OPTIMIZING BACKOFF PROCEDURE FOR ENHANCED THROUGHPUT AND FAIRNESS IN WIRELESS LANS

BY

ADEL ABDULAZIZ AL-AKEEL

A Thesis Presented to the

DEANSHIP OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

COMPUTER NETWORKS

Jumada I, 1428 AH
June 2007

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN 31261, SAUDI ARABIA

DEANSHIP OF GRADUATE STUDIES

This thesis, written by Adel AbdulAziz Al-Akeel under the direction of his thesis advisor and approved by his thesis committee, has been presented to and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of MASTER OF SCIENCE IN COMPUTER NETWORKS.

Thesis Committee

Thesis Advisor

Member

Member

Member

Member

Department Chairman

Dean of Graduate Studies

13/6/09

Date

ACKNOWLEDGMENTS

My unreserved praise and gratitude are for Allah, the Most Gracious, the Most Compassionate, and the Most Merciful. He blessed me with his bounties. May his peace and blessing be upon our prophet Muhammad, and his family.

I would like to express my deep and lasting gratitude to my advisor, Dr. Ashraf S. Hasan Mahmoud. It was my luck to work closely with him thereby benefit from his expertise in network modeling and simulation. I am greatly indebt to Dr. Ashraf for the guidance, support and time offered to me over the past years during my work in this research.

Dr. Uthman Baroudi and all my committee members, Dr. El-Sayed El-Alfy, Dr. Mayez Al-Mouhamed, Dr. AbdulAziz Al-Mulhem and Dr. Adnan Gutub, have been very helpful in improving my proposal and thesis. I am grateful to them for sharing their time and expertise.

It is my pleasure to acknowledge the support and facilities provided by the Computer Engineering Department at King Fahd University of Petroleum and Minerals. An equal acknowledgment goes to the College of Computer at Qassim University (located in Qassim, 300 km to the north of Riyadh) for their support.

My parents, my wife and my family have always been supportive of my educational endeavors. I am grateful to them for their support and assistance during my work towards this thesis.

Contents

List of Tables

List of Figures

Abstract

The distributed coordination function (DCF) of the IEEE 802.11 standard for wireless LANs has been the subject of extensive research in recent years due to its popularity and simplicity. However, DCF is known for its low efficiency if operating in networks with high loads. This is a straightforward consequence of the contention algorithm used by the DCF function. The throughput and delay are both affected by this degradation in efficiency.

Enhancements to the DCF have been proposed in the literature. These enhancements optimize one metric at the expense of others. For example, some algorithms enhance throughput at the expense of delay. Other algorithms concentrate on fairness while neglecting performance in high loaded situations.

In our present study, we use simulations to evaluate the functionality of the DCF function in terms of throughput, delay and fairness. Also, we evaluate some of the enhancements to the DCF that appear in the literature. Our main objective is to develop new algorithms that enhance the contention algorithm of the DCF. This is done mainly by enhancing and controlling the backoff procedure of the DCF. Our new algorithms achieve higher utilization of the network in both low and high loaded situations. At the same time, the fairness among stations in the network is maintained at higher levels.

ملخص الرسالة

الاســـــــــم: عادل بن عبدالعزيز العقيل

عنوان الرسالة: تطوير أداء الشبكات المحلية اللاسلكية عن طريق تحسين خوارزميات التغلب على أخطاء التراسلات

التخصـــــص: شبكات الحاسب الآلي

تاريخ التـخرج: جمادى الأولى 1428هـ / يونيو 2007

كثرت الدراسات في الآونة الأخيرة حول الخوارزمية المتشعبة-التحكم (DCF) في النظام القياسي IEEE 802.11 للشبكات المحلية اللاسلكية، وذلك لشعبيتها وسهولة استخدامها. لكن DCF مع ذلك فيها بعض الأخطاء والقصور. ومن ذلك، أن أدائها وكفاءتها تقل عندما يكون عدد الأجهزة (أو التراسلات) كثير في الشبكة. وهذا الشيء راجع إلى نظام التجاذب الموجود في DCF (وهو المكون الأساس لها). ومن صور تدني الأداء بطء التراسلات داخل الشبكة وكذلك تدني مستوى الاستفادة من مكونات الشبكة.

لقد تم اقتراح الكثير من المحاولات لتطوير أداء الـDCF في أبحاث سابقة. هذه المحاولات تحاول تحسين أداء DCF من جانب معياري واحد دون النظر إلى المعايير الأخرى لقياس الأداء. فعلى سبيل المثال، بعض المحاولات نجحت في تطوير مستوى الأداء من ناحية كمية البيانات التي تستطيع الشبكة استيعابها ولكن كان ذلك على حساب الزيادة في بطء عمل الشبكة ونقلها للبيانات. هناك محاولات أخرى ركزت على إعطاء كل جهاز كفايته من كمية البيانات مع التوزيع المتوازن بينها ولكن مع إغفال مستوى أداء الشبكة في حالة وجود ضغط عليها (سواء بوجود عدد كبير من الأجهزة أو وجود عدد كبير من التراسلات).

قمنا في هذه الدراسة بفحص عمل الـDCF من ناحية الكفاءة وسرعة نقل البيانات و العدل بين الأجهزة في استغلال مكونات الشبكة. وقد كان هذا الفحص وما تلاه من التجارب مبنياً على محاكاة عمل الشبكة باستخدام برنامج أعددناه لهذا الغرض. كذلك قمنا بفحص أداء بعض محاولات التطوير المستفادة من أبحاث أخرى. لقد كان هدفنا الأساس في هذه الدراسة هو إيجاد وتطوير خوارزميات جديدة (مبنية على الـDCF) نستطيع من خلالها تطوير أداء الـDCF. واستطعنا الحصول على هذا الهدف من خلال التحكم بنظام التغلب على أخطاء التراسلات الموجود في DCF. باستخدام هذه الخوارزميات استطعنا تطوير أداء الشبكات المحلية اللاسلكية في جميع الظروف، سواء كان عدد الأجهزة المتواجدة في الشبكة قليل أو كثير وسواء كانت كمية البيانات المتبادلة بين الأجهزة قليلة أو كثيرة. وفي نفس الوقت استطعنا الحفاظ على مستوى عال من التوزيع المتوازن بين الأجهزة من ناحية كمية البيانات المرسلة بواسطة كل جهاز والوقت المتاح لكل جهاز لإرسال البيانات.

CHAPTER 1

INTRODUCTION

Wireless communications have spread in many countries in the last few years. This was lead by the cellular wireless communication systems. Cellular systems are mainly concerned with voice. Wireless data communication is relatively new compared to voice services (the first cellular system was deployed in the early 1980s while the first WLAN product was announced in 1990).

Wireless data communications (or mobile computing) can be divided mainly into two categories: those carried over cellular networks (such as GPRS) and wireless local area networks (WLANs).

The IEEE 802.11 is the most popular standard for WLANs today. It defines the functional aspects of the medium access control (MAC) sublayer and the physical layer (PHY) specifications [1]. Two access mechanisms are supported in the MAC sublayer of the IEEE 802.11: the distributed coordination function (DCF) which is distributed and contention based, and the point coordination function (PCF) which is centralized and polling based. More details about these functions will be given in CHAPTER 2.

The IEEE 802.11 standard is the most widely deployed standard for WLANs because of its simplicity, flexibility and cost effectiveness [2]. It can easily be implemented on a chip and IEEE 802.11-based WLANs can easily be deployed without a

special setup. The cost of these WLANs has dropped for both the access points (APs) and the interface cards (most new laptops have the functionality of WLAN built in).

In spite of the above characteristics, the IEEE 802.11 standard has some limitations. Its main drawback is throughput and delay degradation in highly loaded situations. These are situations where the number of stations in the network is large and/or the load generated by each active station in the network is high. This disadvantage is especially apparent in the DCF function of the standard, which is more popular in the market nowadays.

Another drawback of the IEEE standard is that it does not support quality of service (QoS). Multimedia applications such as voice and video require some QoS guarantees. These applications have some requirements on bandwidth, delay, delay jitter and loss rate. The IEEE 802.11 standard supports only best effort services, where there are no QoS guarantees. The PCF function supports polling where each station is granted access to the medium and can utilize it fully. However, this does not ensure a strict QoS guarantee. The limitations of IEEE 802.11 are discussed in more detail in Section 2.2.

There has been much research in the literature to enhance the IEEE 802.11 standard. Different approaches have been proposed. Some of them consider PCF while others consider DCF, which is more widely deployed. These enhancements can be classified into four categories [3]:

- Protocol enhancement mechanisms that improve the performance of the network.

- Service differentiation mechanisms to differentiate between different stations or flows carrying applications with different QoS requirements

- Admission control and bandwidth reservation

- Link adaptation

Our present research concentrates on the first category, and it shows how to enhance the functionality in order to improve the performance of the network by focusing on the important metrics such as throughput, delay and fairness. One of our proposed algorithms considers differentiation as well.

Our solutions study the network in both saturation and non-saturation conditions. Here saturation means that stations always have packets ready for transmission and the buffer is never empty. In the non-saturation case, the stations receive packets from the upper layer according to some arrival process. We give a detailed comparison between the two traffic approaches in Section 3.1.

CHAPTER 2

BACKGROUND AND LITERATURE REVIEW

2.1 Background of 802.11 WLANs

The IEEE 802.11 standard defines the functional aspects of the medium access control (MAC) sublayer and the physical layer (PHY) specifications [1]. This standard defines three physical layer implementations: direct-sequence spread spectrum (DSSS), frequency hopping spread spectrum (FHSS) and infrared (IR). The stations (STAs) in the IEEE 802.11 WLAN can be grouped in one of two configurations: infrastructure and ad hoc. Each group of stations communicating directly with each other belongs to the same basic service set (BSS). The BSS of the infrastructure configuration consists of an access point (AP) and the communicating STAs. In the ad hoc configuration, the BSS − also called independent BSS (IBSS) − does not require an AP.

The MAC specification defines the way stations access the channel (medium). There are two modes for channel access: distributed coordination function (DCF) and point coordination function (PCF). DCF is the main access mechanism and it is mandatory in the standard. It is distributed (no central control on network operation) and contention-based. It can be used with both the infrastructure and the ad hoc

configurations. The PCF is built on top of DCF as shown in Figure 1. PCF is a contention-free and centralized access method which is based on polling. PCF is optional and used only on infrastructure configurations.

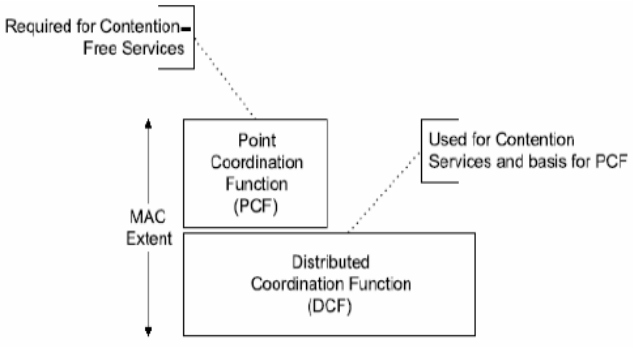

Figure 1: MAC architecture of IEEE 802.11 [1]

To prioritize the access to the medium among stations that want to transmit, the IEEE 802.11 standard defines three main interframe spaces, namely: DCF interframe space (DIFS), PCF interframe space (PIFS) and short interframe space (SIFS). The shortest is the SIFS and it is used for acknowledgements (ACKs) and CTS. The PIFS is used for the point coordinator (PC) when it sends the beacon. The longest, which is the DIFS, is used for regular access in the contention period.

Following is a brief explanation on how stations use DCF and PCF to access the medium.

2.1.1 DCF (CSMA/CA)

DCF is a contention-based access scheme. It is also known as carrier sense multiple access with collision avoidance (CSMA/CA) or listen before talk. This scheme works as follows. When a station wants to transmit, it senses the medium; if it is idle for a period of time equal to DIFS, it transmits the frame; otherwise, it defers and waits until the medium

becomes idle again for DIFS. It then starts the backoff procedure where it will chose a uniformly random number in the range [0,CW-1], where CW is the contention window. The backoff interval is calculated as follows [1]:

$$\text{Backoff Time} = \text{Random}() * aSlotTime \quad (2.1)$$

where *aSlotTime* is the time for an empty slot; it is physical-layer dependent.

During the backoff interval, if the medium is sensed idle for a slot time, the backoff counter is decremented by one. If, on the other hand, the medium is sensed busy, the station freezes (suspends) the counter; when the medium becomes idle again for DIFS, the countdown is resumed. When the counter reaches zero, the station transmits the frame. The backoff procedure is invoked between every two successive packets.

The CW is assigned a minimum value, CW_{min}, at the beginning of the backoff procedure. Its value will be doubled after each unsuccessful transmission; i.e. $CW_i = CW_{min} * 2^i$, where i is the stage of the backoff procedure (the number of successive collisions that occurred during this invocation of the backoff procedure). The CW is increased until the stage m is reached; equally stated, until CW reaches CW_{max}. It remains in this stage until the frame is successfully transmitted or the retransmission (retry) limit is reached. After each successful transmission, the CW is reset to the minimum value, CW_{min}. CW_{min} and CW_{max} are fixed values; they depend on the physical layer (PHY) implementation. For example, CW_{min} and CW_{max} for DSSS are 31 and 1023 respectively while their values for FHSS are 15 and 1023 respectively.

DCF defines two modes of operation: basic access and request-to-send/clear-to-send (RTS/CTS). In the basic access mode, after the source STA contends for the channel and

gains access, it transmits its frame. Following the frame reception, the destination station waits for a SIFS period and then sends a short acknowledgement (ACK) control frame to the source STA. The ACK frame gives the source STA an indication that the frame was successfully received by the destination STA. If the ACK frame is not received within a specified time limit, the source STA assumes that the data frame was not correctly received by the destination STA. In this case, the source STA will schedule a retransmission (if the retry limit is not reached) with the CW doubled as explained above. The working of the basic access scheme is clarified in Figure 2.

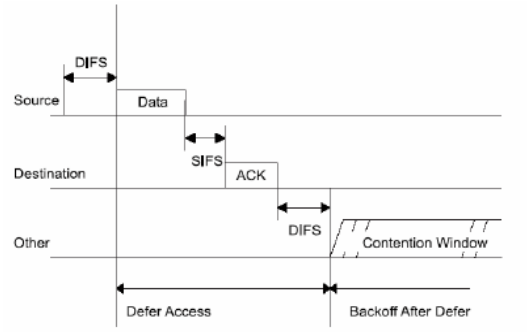

Figure 2: Basic access method of DCF [1]

For RTS/CTS access scheme, special control frames support the actual transmission of data frames. The source station first transmits a short RTS frame to the destination station. A SIFS period following the reception of the RTS frame, the destination station replies with another short frame called CTS. If this CTS frame is correctly received by the source station, it transmits its actual data frame after SIFS period. Finally, the destination station transmits an ACK frame indicating successful reception of the data frame. Because the RTS/CTS scheme incorporates these four transmissions, it is sometimes called “four-way handshake”. The advantage of RTS/CTS is twofold. First, the time of collision (if it occurs) is minimized. Second, the hidden terminal problem is solved. This problem occurs

when there is a station that can hear only one of the communicating stations, and not both. Figure 3 shows the RTS/CTS mechanism.

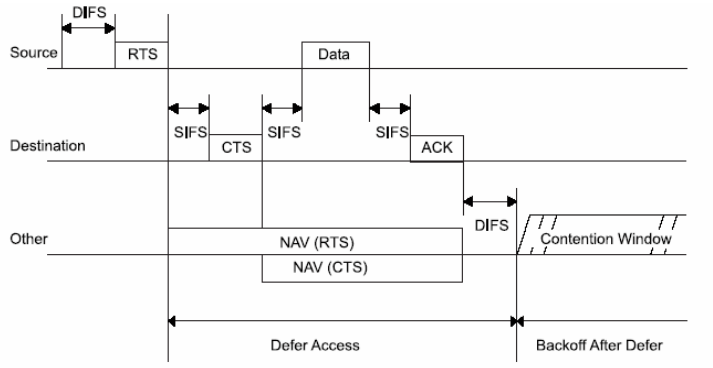

Figure 3: RTS/CTS mechanism [1]

2.1.2 PCF

Although enhancing the PCF functionality is outside the scope of this thesis, we briefly give an insight about PCF and some of its problems (2.2.2) just to complete the picture of the 802.11 MAC. As shown in Figure 1, PCF is built on top of DCF. PCF is a centralized polling-based access scheme. The point coordinator (PC) coordinates access to the medium by issuing polls to stations that are using this access scheme. Usually, the PC resides in the AP. The PC maintains a list of the stations that are 'pollable'. This list is called the polling list. Time axis in PCF is divided into superframes of the same length. These superframes repeat periodically; the period of the superframe is called contention-free repetition interval or CFP_Rep. The superframe is divided into two parts: contention-free period (CFP) and contention period (CP). The CFP starts with a beacon frame transmitted by the AP. During the CFP, the AP polls each station in the polling list with a CF-Poll frame, at least once. The polled station should respond after SIFS period with a CF-ACK frame. If the station has data to send, it piggybacks the data with the CF-ACK.

A SIFS period following the reception of the station response, the AP polls the next pollable station. If the AP does not receive a response from a station within PIFS, it polls the next station in the polling list. This procedure continues until all stations in the polling list are polled or the CFP_Max_Duration (CFP_Max) is reached. The AP then sends a CF-End frame to indicate the end of the CFP and the beginning of the CP. During the CP, stations contend for access to the channel in the same way as they do in DCF. A new superframe starts when the AP sends the beacon at the target beacon transmission time (TBTT). Figure 4 illustrates the functionality of PCF.

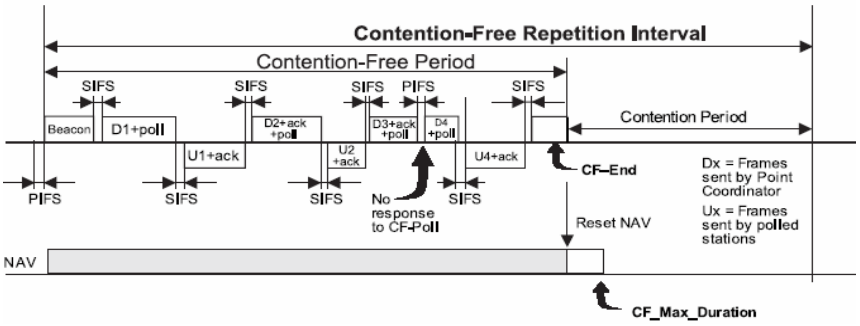

Figure 4: PCF mode [1]

2.2 Limitations of IEEE 802.11

In this Section we briefly show some of the limitations of the current standard in terms of performance and functionality. Both DCF and PCF have some problems in this regard. PCF has better performance than DCF in terms of throughput and delay [6]. Nevertheless, it has some limitations as shown in Section 2.2.2. Detailed comparison between DCF and PCF is outside the scope of this research.

2.2.1 Limitations of DCF

DCF is a contention-based access scheme, and thus is not suitable for time-bounded applications such as voice or video conferencing. Typically, these applications are associated with specific bandwidth, delay and delay jitter constraints. Using the DCF procedure, there is no service differentiation mechanism to allow such applications to meet their QoS requirements. More importantly, as with most contention-based methods, the media access delay is not controllable. For example, when a collision occurs due to multiple transmissions from different stations, the backoff procedure is invoked, with a progressively increasing backoff window. This results in further delays for the frame in hand that is going to be transmitted.

The channel utilization in DCF is low. This is true in two situations: low loaded and high loaded (although much clearer in high loaded situations). In low loaded situations, the bandwidth is wasted by waiting for many time slots before transmitting because of the invocation of the backoff procedure. The backoff procedure is invoked between every two successive transmissions (i.e. two packets). In high loaded situations, the bandwidth is wasted by multiple collisions. We will show later that collisions are a major factor in degrading throughput and delay. Actually, collisions have more effect on these metrics than empty slots.

In the DCF function of the 802.11 standard, the CW is doubled following each collision. Furthermore, the CW is set back to CW_{min} following each successful transmission. This has two disadvantages. First, due to the former, fairness is not guaranteed. This is because, if a transmission from a station collides, then the CW is doubled and the station needs to wait more to retransmit. In this period, if a station with a

new frame attempts to transmit, it will choose a CW set to the minimum size, CW_{min}, and on average will succeed before the frames that are initiated from other stations already in the backoff process although its frame was scheduled for transmission later.

The second disadvantage of conventional DCF functionality is the congestion in the network in high loaded situations. This is a straightforward result of reducing the CW size to its minimum following a successful transmission. For example, suppose that there are 30 active stations in the network. If there is a collision between three or four frames, and if this collision occurred while the stations initiating these transmissions are in the second stage of the backoff procedure or above, then all these stations will double their CWs. If they eventually succeed, they will reduce their CW to its minimum. In this case, a new collision will occur with very high probability, taking into account that there are other stations having frames to be transmitted.

Another main problem that usually occurs, and is a main factor in degrading the throughput of DCF, is the multiple collisions in high loaded situations. The DCF deals with collisions as follows. The backoff procedure is invoked and the CW is increased exponentially; i.e. doubled with each collision. Until the time the CW reaches the suitable size where collisions are minimized, multiple collisions occur, causing the throughput to degrade. In CHAPTER 4 and CHAPTER 5 we will present our solutions to these problems and show the results of applying these solutions.

2.2.2 Limitations of PCF

As explained in Section 2.1.2, PCF is centralized in nature, which requires the PC to control traffic in the network. One of the drawbacks of PCF is the following. The inter-

BSS traffic (i.e. traffic between two stations in the same BSS) has to go through the AP. This introduces more overhead on the channel [2] and will adversely affect throughput and delay of the system, especially in high load scenarios.

In contrast to CFP procedures, a station that wants to send a frame in the CP period does not check for time availability before the time for the next beacon (maybe the reason for this is to keep DCF as simple as possible). As a consequence, the transmission of the beacon at the start of the CFP may be delayed due to the channel being busy with a DCF transmission. This has an impact on time-bounded applications that must meet specific QoS constraints. In the worst case, the maximum delay for the beacon frame is 4.9 milliseconds in IEEE 802.11a [4][5] while the average beacon delay is 250 microseconds [5].

The transmission of a polled station is not controllable. A polled station may have a variable frame length (0-2340). Furthermore, the transmission rate of the polled station is not predictable in advance. This may introduce latencies for other stations in the polling list, which degrades the QoS performance of PCF [2]. As per the IEEE 802.11 standard, the AP should poll each station in the polling list at least once in each CFP. This further introduces an overhead (and wastes bandwidth) if some stations do not have frames to send.

2.3 Literature Review

In this section we give an overview about the recent and most important activity in the literature that is concerned with enhancing the DCF function of the IEEE 802.11 standard. This issue is studied with details in the literature. Much of these studies are simulations, while some of them analytically study the standard and its versions.

In [6] the author developed an analytical model for the DCF function to study its two methods: basic access and RTS/CTS. The model is used to evaluate the throughput of the network. The study utilizes a bi-dimensional Markov chain model (Figure 5) for the backoff window size. From this model, the throughput of the system is calculated with some approximations. The author showed that the performance of the basic access scheme, in terms of throughput, strongly depends on the network size (number of stations in the network), the initial window size (CW_{min}) and the number of stages of the backoff algorithm. For the case of RTS/CTS, the effect of these parameters has been found to be marginal. The study also examined the effect of the packet size on the system performance, and it calculated a packet size threshold over which the RTS/CTS access mechanism is more efficient. The study concluded that RTS/CTS mechanism is superior, in terms of throughput, in most of the cases.

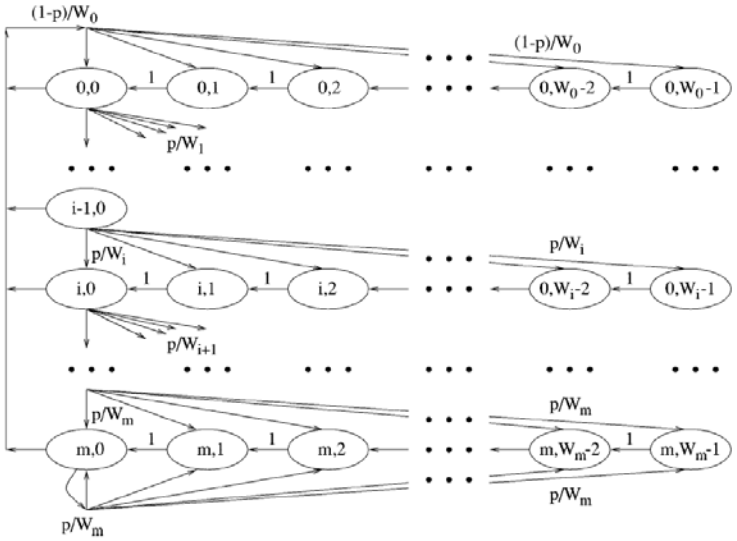

Figure 5: Markov chain model for the backoff window size of the DCF function [6]

Ziouva and Antonakopoulos in [7] extended the analytical model developed in [6], with some refinements, to evaluate the frame delay as a function of system parameters. The authors showed that the CW_{min} and the number of backoff stages (in addition to the network size) affect the delay of both the basic access and RTS/CTS mechanisms.

Based on the model used in [6] and [7], the study in [8] derived an analytical model for a prioritization scheme. The introduced priority scheme is based on differentiating three parameters: the initial window size (CW_{min}), the window increase factor, and the maximum backoff stage. The effect of differentiating these parameters on network throughput and delay is studied by using a traffic mixture with two priority classes. The numerical results presented show the effect of each prioritization parameter.

In [9] and [10], various differentiation mechanisms on per-terminal were introduced; namely, backoff differentiation, DIFS differentiation, maximum frame length differentiation and CW_{min} differentiation. In the backoff differentiation mechanism, each

terminal increases its backoff window (CW) by a factor P_i. With different factors $P_i \neq P_j$ for different terminals *i* and *j*, different priority classes (one for each terminal) are obtained. In the second scheme, DIFS differentiation, each terminal waits for different IFS before sending; lower priority classes wait longer and vice versa. For the third scheme, the flow of different priority classes (terminals) is differentiated by limiting the maximum frame size allowable for each terminal. In the last scheme, differentiation is based on CW_{min} where higher priority classes have lower value of CW_{min} and vice versa.

The results of the simulations conducted in [9] and [10] show that backoff differentiation-based and CW_{min} differentiation-based work well with UDP flows but do not have differentiation effects on TCP flows. The DIFS differentiation-based and the maximum frame differentiation-based mechanisms are suitable for both UDP and TCP flows. Furthermore, if operating in a noisy environment, the backoff scheme and the maximum frame size scheme perform poorly compared to the case of DIFS differentiation, which suffers little or no impact. The effect of per-destination differentiation to solve the problem of TCP flows was studied in [10]. This mechanism showed no improvement, and no clear differentiation was achieved. As another solution to this problem, per-flow differentiation was suggested, where each flow is assigned a separate queue; however, no evaluation was done. The study concluded that the DIFS differentiation mechanism is the best among the ones studied.

Zhao and Fan [11] introduced a new method (criteria) for differentiating traffic in WLANs. The new approach, called integrated QoS Differentiation (IQD), considered packet loss rate as a means of differentiation. This was achieved by assigning different retry limits for different traffic categories. An analytical model was developed on the

basis of [6] and simulations have been conducted. The results (of both the analysis and simulation) showed that IQD achieves good differentiation in the context of packet loss in addition to delay and throughput differentiation.

In [12] and [13], Campbell et al. studied differentiation through changing CW_{min} and CW_{max}. By using CW_{min}, delay differentiation can be achieved; by using CW_{max}, packet loss probability can be achieved (this is questionable because CW_{max} does not control the dropping of the frame; the retry limit does so as in [11]).

In the IEEE 802.11 standard, the CW is reduced to CW_{min} after each successful transmission and doubled after each collision, as discussed in Section 2.2.1. This results in more collisions following a successful transmission and thus throughput is degraded. Cali et al. [14][15][16] introduced an adaptive scheme for setting the size of the CW. They first analytically derived the maximum throughput that can be achieved by an IEEE 802.11 WLAN (the theoretical limit). It is claimed that the throughput achieved when using the new adaptive scheme is always close to the theoretical limit. In this distributed scheme, a station estimates the status of the network (collision cost and number of active stations) by observing idle slots, collisions and successful transmissions. These estimates are updated after each successful or erroneous transmission. Based on these estimates, the optimal CW size is chosen. The advantages of the algorithm were verified via simulations. Although this algorithm achieves good results, it is complex to implement in a WLAN station. The existing 802.11 DCF stations need some modifications to implement this algorithm; i.e. the algorithm is not backward-compatible with the legacy standard. Actually, in this new algorithm, the optimal CW size is not chosen from a uniform distribution; rather, it is chosen from a geometric distribution with parameter *p*.

Furthermore, the time it takes a station to learn the new number of stations in the network, referred to therein as the convergence time, is relatively long (up to 40 seconds to react to the change from 2 to 10 stations).

In a similar (but simpler) manner, Ni et al. [17] proposed a new scheme, named CW slow decrease. The new scheme reduces the CW to a value near the old one following a successful transmission. The CW is decreased according to some decrease factor that is evaluated during run time. This would reduce collision probability when the network is congested. Eventually, throughput is improved. This was analytically shown in [17] with a model based on the one in [6]. The numerical results showed that the throughput is better when using CW slow decrease, especially when the number of stations is large and CW_{min} is small.

Wang et al. generalized the scheme in [17] (as part of the work in [18]) to study the performance of CW slow decrease under multiple priority flows. This would provide QoS differentiation among these flows. The new scheme, called gentle DCF (GDCF), counts the number of consecutive successful transmissions. If c of them occur (c is a network design parameter), the CW is halved. By varying c for different flows, differentiation can be achieved. Obviously, higher priority flows are assigned smaller values of c.

In [19] and [20], Sobrinho and Krishnakumar proposed a new contention-based distributed algorithm to support real-time traffic over WLANs. The new scheme, named black burst (BB), does not require changes to the MAC sublayer of IEEE 802.11 except that real-time terminals must be able to jam the channel for specific intervals of time. The scheme works as follows.

At the first attempt to transmit a frame (start of a session), the real-time station transmits its frame according to CSMA/CA procedure. At that time, it schedules its next attempt to access the channel, say after t_{sch} (it does so from the first transmission onwards). After t_{sch}, if the channel is idle for PIFS, the frame is transmitted. Otherwise, the station defers until the channel is idle for PIFS, where it sends its black burst (BB). The length of BB is proportional to the access delay encountered from the attempt to access the channel until BB is sent; it is expressed in black slots. After sending BB, the station waits an interval t_{obs}, to see if another station is sending a longer BB, which implies it was waiting for a longer period. If the channel is idle for t_{obs}, the station transmits its frame; otherwise, it defers again until the channel is idle for PIFS where it sends another longer BB that reflects the increase in access delay. If t_{sch} is fixed for all real-time stations, the algorithm guarantees that there is a unique winner in each BB contention. Otherwise, the algorithm needs to be enhanced to accommodate real-time sessions with different (finite and small) scheduling intervals.

The authors analyzed and verified the working of the algorithm mathematically and by simulations. The results of the simulations showed that BB outperforms CSMA/CA. In BB, the number of real-time stations accommodated was greater. Also, the delay and jitter of both data and real-time traffic were reduced when the BB access mechanism was used.

In [21], the authors developed a queuing model to emulate the working of IEEE 802.11 DCF under non-saturation conditions. By using this model, expressions for throughput and delay were derived. They assumed ON/OFF arrival process with geometrically distributed message sizes (the message size determines the length of the ON-period). They verified the accuracy of the model by simulation. Although this model

does not translate the details of the standard DCF in an exact manner, it sufficiently evaluates the performance of the DCF under non-saturation conditions.

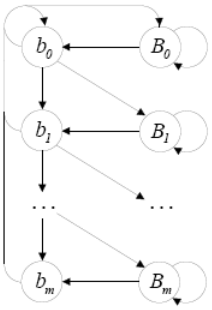

Figure 6: Markov chain model for DCF [22]

A more interesting and comprehensive performance evaluation study of the 802.11 DCF under non-saturation conditions was done in [22]. The study first evaluated the network under saturation conditions. A Markov chain model represents the system as shown in Figure 6. This model is similar to, but simpler than, the one in [6]. The results are the same as in [6]. For the non-saturation condition, two models were developed to study the performance of the network in terms of number of packets queued, the queuing delay and the packet loss probability. A Poisson arrival process was applied to the two models. The first model showed that if operating in a non-saturation condition, the 802.11 network cannot be correctly analyzed relying on the independence assumption among stations. The second model is more accurate but slower to get the results. The authors also examined the performance while applying packet arrivals other than Poisson. To approximate the traffic produced by TCP or sporadic on-off traffic sources, the authors assumed a Poisson batched arrival process. They showed that the burstiness of data arrival increases the average queue length (and hence queuing delay) if operating below saturation.

Regarding the studies that discussed and evaluated fairness in WLANs, an early attempt was done in [23]. The authors studied the fairness of wireless networks that implement the CSMA/CA WaveLAN standard. They used two methods to quantitatively measure the fairness of WaveLAN and compare it with the fairness of slotted ALOHA; the first is the sliding window method with the Jain fairness index and Kullback-Leibler distance, and the other is the renewal rewards method. It was shown that the WaveLAN networks suffer from short-term unfairness. Although this study introduced new methods for evaluating fairness in wireless networks, it did not measure the fairness of the DCF function of the standard IEEE 802.11 WLANs. There are significant differences between the WaveLAN and the IEEE 802.11. The access method is similar but the backoff procedure is different. In the WaveLAN, the backoff procedure is invoked when the channel is sensed busy. However, in the standard DCF function, it is invoked when a collision occurs. Also, in the WaveLAN standard, the station does not suspend the count down while other stations are transmitting.

This difference was clarified in [24] showing the correct result of the fairness of the DCF of the standard IEEE 802.11 WLANs. A new fairness index was proposed therein; namely, the number of inter-transmissions that other hosts may perform between two transmissions of a given host. The probability distribution of the number of inter-transmissions is derived to measure the fairness of DCF for the case of two stations. By using the results of this analytical study, along with simulations and experimental measurements, it was shown that the DCF does not suffer short-term unfairness. The simulations and experiments were also applied for more than two stations, and the sliding window method with the Jain fairness index was also used.

A new fair-share scheduling algorithm, based on weights of flows, was introduced in [25]. This algorithm, named Distributed Fair Scheduling (DFS), is based on the Self-Clocked Fair Queuing (SCFQ) algorithm [26]. The DFS algorithm achieves differentiation between flows by varying the backoff window. The backoff interval for each flow is directly proportional to the frame size and inversely proportional to the weight of the flow; hence maintaining a smaller window for higher priority flows.

To improve channel utilization, the backoff interval is recalculated according to some function (exponential or square root [25]) while maintaining fairness between flows. The results showed that DFS achieves better throughput and fairness over DCF in IEEE 802.11.

Another fair-share scheduling algorithm was introduced in [27]. This algorithm is based on weighted fair queuing [28]. The Distributed Weighted Fair Queuing (DWFQ) algorithm allocates bandwidth to each station or flow proportional to its weight (as in DFS). The bandwidth allocation is achieved by setting the CW to the appropriate value. The computation of the CW is dynamic, where it is updated each time a frame is transmitted. In overloaded situations, the computation of CW is slightly different than that in normal situations so that the throughput is not harmfully affected.

CHAPTER 3

SIMULATION SETUP

3.1 Simulation Parameters and Setup

For the evaluation, we employ a flexible simulation tool that is developed under MATLAB. Some assumptions to simplify the process are made. This is a common practice in running simulations since we are interested in specific metrics and factors that affect the network. We assume the following:

- error free channel
- no RTS/CTS
- no hidden terminals
- modulation used is DSSS
- Bit rate is 1 Mbits/sec

The parameters used in the simulation are summarized in Table I. These are the default values used throughout the simulations unless otherwise stated.

TABLE I: Simulation Parameters

Parameter	Value	Parameter	Value
Packet Payload	8224 bits	Channel Bit Rate	1 Mbps
MAC Header	224 bits	Propagation Delay	1 μs
PHY Header	192 bits	Slot Time	20 μs
ACK	112 bits + PHY Header	SIFS	10 μs
CW_{min}(slots)	31	DIFS	50 μs
CW_{max}(slots)	1023	Retry Limit	7

These parameters and ideal conditions are usually found in the literature to evaluate and enhance the backoff procedure of the IEEE 802.11 DCF function. For example, a very important study in this field assumed the same assumptions [6]. Other studies also adopted similar parameters.

The main parameters that affect the performance of the algorithms that are going to be studied are q (a system parameter in the q algorithm), the number of stations in the network (n), CW_{min}, CW_{max} and frame size. We will show in our simulations the effect of each of these parameters.

The scenario of our simulations is as follows. We simulate a WLAN network that consists of n wireless stations (STA) and one access point. All the communication is destined to the access point; it serves as a sink for the data. The access point is connected to the wired part of the network and the bandwidth of this connection is large enough to ensure that the bandwidth bottleneck is in the wireless network. The number of stations in

the network is fixed in any simulation run. Also, all stations are located in the transmission range of each other; i.e. each station can detect the transmission of any other station. This prevents the hidden terminal problem. Moreover, there is no mobility within the network.

We use two kinds of traffic in our simulations. The first one reflects a saturation state in the network. This means that each station has a frame ready to be transmitted following any transmission. In other words, the transmission queue (the queue that receives frames from higher layers) is never empty. The second kind of traffic is an arrival process that delivers frames to the stations according to a Poisson distribution. That is, the interarrival time between frames that are received from higher layers is exponentially distributed.

Using the first kind of traffic, the worst case scenario of the network can be studied. This helps to establish some regulations or to detect the higher capacity of a network in any condition. Also it helps to establish the upper limit in congested places where the network is to be implemented. On the other hand, the second kind of traffic is much closer to the actual network in the real world. Also, simulating different loads enables one to differentiate between different sources of information. For example, the load for data differs from the load for voice or video.

3.2 Performance Metrics

Our goal in this work is to evaluate and enhance the functionality of the DCF function of the IEEE 802.11 networks. As shown in Section 2.2.1, the DCF suffers from many limitations such as throughput and delay degradation in high loaded situations. In this Section, we define the performance metrics that we are going to evaluate. These performance figures are: throughput, delay and fairness.

The throughput of the network is the main metric in most of the networks that are studied in the literature. It gives an indication about the effectiveness of the protocol in hand. Also, it gives an overview about the utilization of the channel in different cases and situations. This in turn helps in making different decisions regarding deployment of the network and maintaining it.

In this thesis, we evaluate the throughput of the network as a measure of the performance of the protocols and algorithms we study. Our method for measuring the throughput is simple. We measure the time spent in transmitting frames that are successfully received, and we divide it by the total time. Namely, we use the following equation for measuring the throughput of the network:

$$\text{Avg\_Throughput} = \frac{\text{Total\_Packets - Dropped\_Packets}}{\text{Max\_Time/Frame\_Time}} \tag{3.1}$$

where:

Total_Packets:	The total number of packets sent during simulation time.
Dropped_Packets:	The total number of packets that are dropped because of reaching the retry limit.
Max_Time:	The total simulation time.
Frame_Time:	The time needed for sending one frame.

This equation gives the average throughput including the time for the overhead bytes. To get the real throughput that gives a measure of the useful transmitted bytes, we apply the following equation:

$$\text{Throughput} = \text{Avg\_Throughput} * \frac{\text{T\_data}}{\text{T\_data+T\_mac\_H+T\_plcp\_H}} \quad (3.2)$$

where:
T_data: Time needed to transmit the actual user data.
T_mac: Time needed to transmit the MAC overhead bytes.
T_plcp: Time needed to transmit the PHY overhead bytes.

The access delay is an important metric in the network, especially for real time applications. The access delay is defined as the time required for a frame to be totally received at the destination station since it was at the head of the transmission queue of the source station. The queuing delay is also an important metric. We define the queuing delay as the time of the frame from being received from the upper layer until it is considered for transmission (being at the head of the queue). The frames that are considered in calculating the access delay and the queuing delay are the frames that are not dropped because of reaching the retry limit. That is, if a frame is dropped because, after several retransmissions, it reached the retry limit, this frame is not considered in the delay calculation. Only frames that are received by the destination (even after several retransmissions) are considered in the calculation of access delay and queuing delay.

The queuing delay helps in deciding the size of the queue used in the wireless station (STA). The importance of this issue is clearer when the frame size is large. Also, it helps, along with the access delay, in determining the limits of the delay and number of stations for real-time applications. There are many time-bounded applications where the

time limit is strict. For example, a voice frame is discarded in some applications if the end-to-end delay exceeds 250 ms. The delay may reach this limit in some situations such as an increasing number of stations. In this case, admission control should be applied. In our present work, we are not checking time limits; however, we are giving delay measures which any application could benefit from. The two traffic sources that we are using can be considered as approximations for any application source.

Fairness is another important metric that influences the overall performance of the network. Fairness can be classified into short-term and long-term [24]. Long-term fairness is observed over long time periods (corresponding to transmission of thousand frames or more). A protocol is said to achieve long-term fairness if each station receives $1/n$ of the total bandwidth, assuming there are n stations contending for access in the network. Short-term fairness, on the other hand, is observed over short time scales (corresponding to transmission of ten frames or less). For a protocol to be considered as short-term fair, each station should transmit briefly, and no station may starve the channel for a longer time.

Both figures (short-term fairness and long-term fairness) are important; long-term fairness gives an indication about the sharing of the channel bandwidth, and short-term fairness is important for a good performance of the applications and protocols. Protocols with short-term unfairness are subject to performance impairments such as increased jitter. Real-time applications such as voice and video are sensitive to jitter and may have serious performance implications if the jitter increases. Also, TCP flows may be affected when the MAC layer protocol exhibits short-term unfairness. The ACK may be delayed

and the congestion window size is not controlled [23][24]. A detailed discussion about this issue can be found in [23] and [24].

Different measures can be used to measure the fairness of a protocol or an algorithm. Examples of these are: the sliding window method [23], the renewal reward theory [23] and the inter-transmissions of other stations [24]. In this research we use the sliding window method with the Jain fairness index [23]. This method works as follows. A packet trace of channel accesses is taken, and a window of size *w* slides across it. Figure 7 shows a trace of transmissions of two sources: A and B with $w = 4$. The first window consists of the sequence AAAB. We slide the window one element at a time to obtain a series of snapshots, where consecutive snapshots have (w-1) elements in common. For each snapshot we compute the fraction of A's and B's and measure the per-snapshot fairness index. We use the Jain fairness index [29] for this purpose which is defined in Equation 3.3.

$$F_J = \frac{\left(\sum_{i=1}^{N} \gamma_i\right)^2}{N \sum_{i=1}^{N} \gamma_i^{\,2}} \tag{3.3}$$

where N is the number of stations

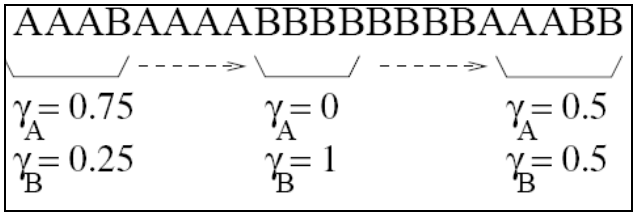

Figure 7: Sliding window method [23]

Formula 3.3 results in a fairness index that lies between 0 and 1. One indicates absolute fairness. After sliding the window through the entire sequence, we end up with a sequence of fairness values. We calculate its average. This average value corresponds to the fairness metric associated with window size *w*. The window size is useless if it is not normalized to the number of stations. So, we normalize the window size as $w = m * n$, m=0,1,2,3,……where m is the normalized window size. Finally, we plot the fairness versus the normalized window size. For each protocol there is a fairness value (threshold) above which the protocol is said to be fair. Usually, this suitable value (or threshold) is 0.95. In the literature, this threshold value is used to compare the fairness of different protocols. If the smallest normalized window that achieves this threshold is small enough, the protocol is said to be short-term fair.

3.2.1 QoS Support

Providing QoS is one of the most challenging problems in wireless networks nowadays. In the context of WLANs, we mean by QoS, the mechanisms or techniques that can provide different priority to different users or applications. One of the methods for enhancing the DCF function of the IEEE 802.11 standard in order to support QoS is to differentiate between different stations or applications according to their needs. In the case of differentiation, there are no guarantees as in many QoS techniques. However, each application type or data flow will receive on average the percentage of throughput that supports its needs. Also, the access delay for some applications or flows would be enhanced (i.e. decreased) but the delay and jitter constraints will not be guaranteed.

Many techniques in the literature are used for differentiation in WLANs. Some of these techniques are found in [2], [8]-[13]. These techniques have some common aspects.

The general method that is used in those reported studies is to control one or more of the DCF parameters as to achieve differentiation. Examples of these parameters are CW_{min}, backoff procedure, DIFS…etc (see Section 2.3). The goal of these techniques is to prioritize access to the channel between different flows as to achieve differentiation.

In our work, we show similar techniques that will help in achieving differentiation between different flows in the network. We did not perform simulation studies that show the results of such work. However, the general idea is explained.

CHAPTER 4

THE *Q* ALGORITHM

Motivated by the need to improve the performance of DCF without the need for difficult-to-tune parameters or extensive training/convergence time as in [14] and [15], we propose two new algorithms that capitalize on the need to reduce congestion especially at high loads. The first algorithm is called *the q algorithm* which we describe in this chapter. The second one is called *the two-stage algorithm* and it is described in the next chapter.

4.1 Description

The *q* algorithm is outlined in Figure 8 and the corresponding flow chart is depicted in Figure 9. The algorithm works as follows. At the beginning, the station will set its CW to its minimum, CW_{min}. The backoff interval, B_i, is uniformly chosen from the range [0,CW) and the station transmits its frame. If there is a collision, the collision counter is incremented. In this stage, the CW is not doubled and B_i is again uniformly chosen from the same range [0, CW). The CW size remains without change and only starts to double, as in the conventional DCF algorithm, when the number of successive collisions reaches *q*, a tuning parameter for our algorithm. After *q* collisions, the operation of the proposed algorithm in terms of extending the CW is identical to that for the conventional DCF. The

retransmission attempts of the frame of interest continue until the frame is successfully transmitted or the retry count reaches its maximum

If there is a successful transmission following q consecutive collisions, then the CW is kept at its value just before the successful transmission, and the collision counter is reset to zero. For the next frame, the CW is set back to its minimum value. The CW is not minimized immediately after a successful frame; instead, it is minimized a frame later.

For example, if $CW_{min} = 8$ and $q = 2$, the CW will not be doubled until after the second collision. Suppose there were two consecutive collisions, then CW would still be 8. If there is another third collision, the CW is doubled to become 16. If a successful transmission occurred at this point, then the CW will not change; i.e. CW = 16. If the next frame is also successful, then at this time only the CW is set to CW_{min} (refer to Figure 8 and Figure 9).

```
Let CW = CWmin
Set collision_counter = 0
Choose Bi uniformly distributed in [0, CW)
Loop:
Transmit Frame
If collision
        If collision_counter < q                (where q is constant)
                Leave CW unchanged
                Choose Bi uniformly distributed in [0, CW)
        Else
                CW = CW * 2
                Choose Bi uniformly distributed in [0, CW)
        Increment collision_counter

If successful transmission
        If collision_counter < q
                Set CW = CWmin
                Choose Bi uniformly distributed in [0, CW)
        Else
                Leave CW unchanged
                Choose Bi uniformly distributed in [0, CW)
        Reset collision_counter to zero
End Loop
```

Figure 8: Proposed q algorithm.

Now, let us consider a special case of the q algorithm. If $q = 0$, we notice that the flow of operation always chooses the second branch of the "if" statement in both collision and success states. In this case, the CW is doubled each time there is a collision (like DCF). However, the CW is never set back to its minimum size, even if there is a successful transmission. Eventually, the CW will reach its maximum size and will not change. This has an advantage and a disadvantage as explained in Section 4.2.

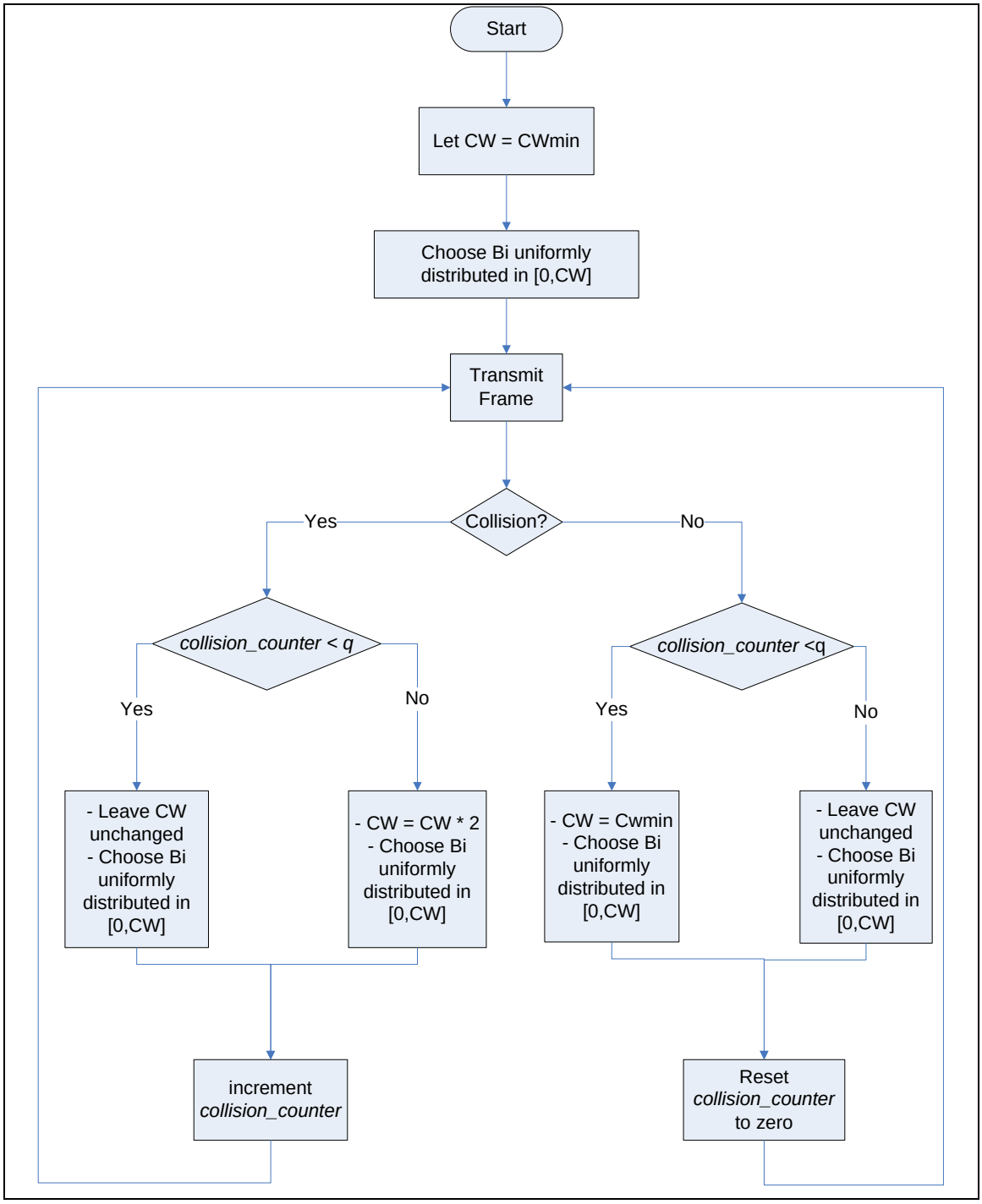

Figure 9: Flow chart of *q* algorithm.

4.2 Results

In this section, we evaluate the proposed q algorithm and provide characterization for its performance in regard to throughput, delay and fairness characteristics. We also compare our results against those obtained for the conventional DCF algorithm.

4.2.1 *q* Algorithm at the Saturation State

We first implement the q algorithm explained above with the first kind of traffic, that is, with the network in the saturation state. In our simulation, we increase the load by gradually increasing the number of stations, n, in steps of ten (except for $n < 10$, we experiment 4 cases).

Figure 10 shows the overall system throughput versus the number of stations (n) with different q. When $q = 1$, 2 or 3, we see that the trend is the same as 802.11 DCF; namely, the throughput decreases when the number of stations increase. For all n, 802.11 DCF performs better than the q algorithm if $q = 3$. However, if $q = 1$, the q algorithm outperforms DCF. For $q = 2$, DCF performs better if $n < 65$; otherwise, the q algorithm's performance is better. For the special case $q = 0$, the throughput of the network is maximum (and always greater than 0.78) if $n > 8$. However, the throughput is as low as 0.6 if $n = 2$.

The reasoning behind this is the following. When the number of stations in the network is small, it is better to choose a small value for the CW. This is what is done by DCF and the q algorithm with $q > 0$. However, if $q = 0$ and if the load is light, the value of the CW increases with successive collisions and never set back to its minimum; hence, the

throughput is wasted with empty slots. In high loaded situations ($n > 10$), DCF quickly sets the CW back to its minimum, CW_{min}, after a single success. This leads to more collisions and hence the throughput decreases. The same thing is done by the q algorithm if $q > 0$ but with slower fashion; i.e. the CW is set to its minimum after two successes if the collision counter exceeds q. However, if $q = 0$, the CW is never set back to its minimum, as explained in Section 0. This results in smaller number of collisions when the number of stations is large, and hence the throughput is improved.

Depending on the number of stations in the network, the throughput can be maximized by choosing the appropriate value of q (see Figure 10). If the number of stations is less than 8, the throughput is maximized if $q = 1$. Otherwise, the throughput is maximum with $q = 0$. Therefore, if a WLAN station can know (or at least estimate) the number of active stations in the network in any given time, then the throughput is guaranteed to be maximized (of course this estimation should be fast enough because the number of stations varies dynamically). This is beyond the scope of our present work and we leave it as a suggestion for future research. If this procedure is to be done (i.e. choosing q dynamically), it is obvious that the new algorithm outperforms DCF in all cases (i.e. for all n). Moreover, when $q = 0$, the improvement in the throughput (compared to DCF) is 0.19 if $n = 30$, 0.24 if $n = 80$ and 0.30 if $n = 120$ (see Figure 10).

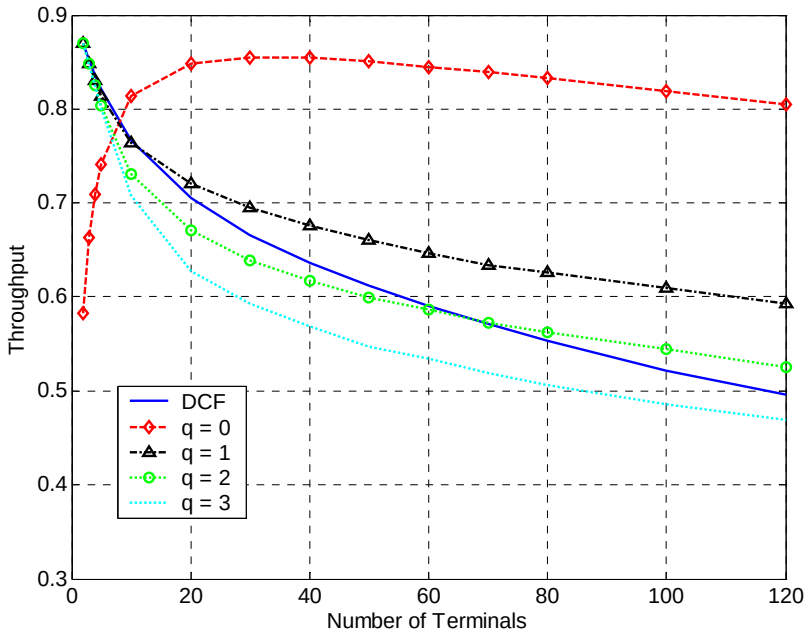

Figure 10: Network saturation throughput vs. number of terminals (DCF and *q* algorithm).

From Figure 11, we see that the delay for the *q* algorithm is less than that for 802.11 DCF if $q = 0$. This is clearer when n becomes large. The higher delay in the case of 802.11 DCF is due to multiple collisions a packet suffers before it is finally transmitted successfully. This is not the case in the *q* algorithm with $q = 0$ since, after a short time, all stations set their CWs to a value where collisions are minimum (this value is CW_{max}, the maximum size of CW). This shows that the wasted time due to multiple collisions (taking into account the frame size, acknowledgement size and inter-frame spacing) is larger than the wasted time due to empty slots. The same conclusion can be found if we concentrate on the case where n is small. That is, if the number of stations is small, we see that the delay of the *q* algorithm with $q = 0$ is the same as that of the DCF although the throughput of the DCF is higher. This shows that, even if the collisions in the case of DCF are few,

their impact on access delay is so high that it is comparable to the impact of very large number of empty slots in the case of the q algorithm with $q = 0$.

Let us take a closer look at Figure 11. If the number of stations in the network is 10, then the average time it takes to transmit a frame in the 802.11 DCF is around 100 msec whereas it takes on average 90 msec to transmit a frame in the q algorithm with the same number of stations. If the number of stations in the network is 50, it takes 475 msec to transmit a frame in the q algorithm, while it takes 555 msec in DCF. If $q > 0$, the delay of the new algorithm is most of the time larger than the delay of DCF (for $q = 1$, this is true only for $n > 30$; otherwise, the new algorithm and DCF perform equally in terms of delay).

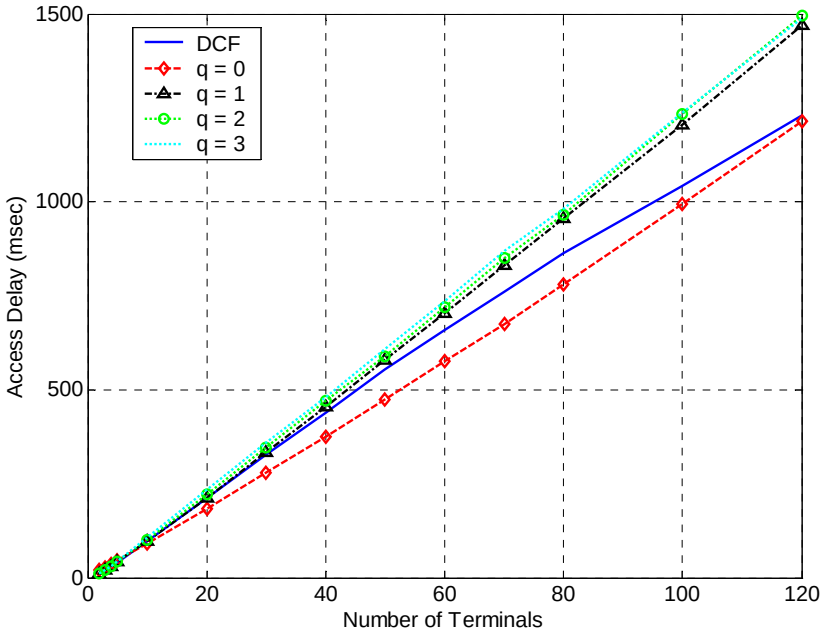

Figure 11: Delay vs. number of terminals (DCF and q algorithm).

From Figure 11, we can see that when $n > 80$, the delay for DCF decreases. This is not because the functionality of the algorithm is superior in this region. However, a great

portion of the transmissions are withdrawn because they reach the retry limit before they are successfully received. As shown in Figure 12, the transmissions that are received by the destination (maybe after successive retransmissions) decrease in number as the number of stations, *n*, increases. Accordingly, the statistics in this region for DCF are not accurate and cannot be trusted for the delay figures after this region (i.e. $n > 80$).

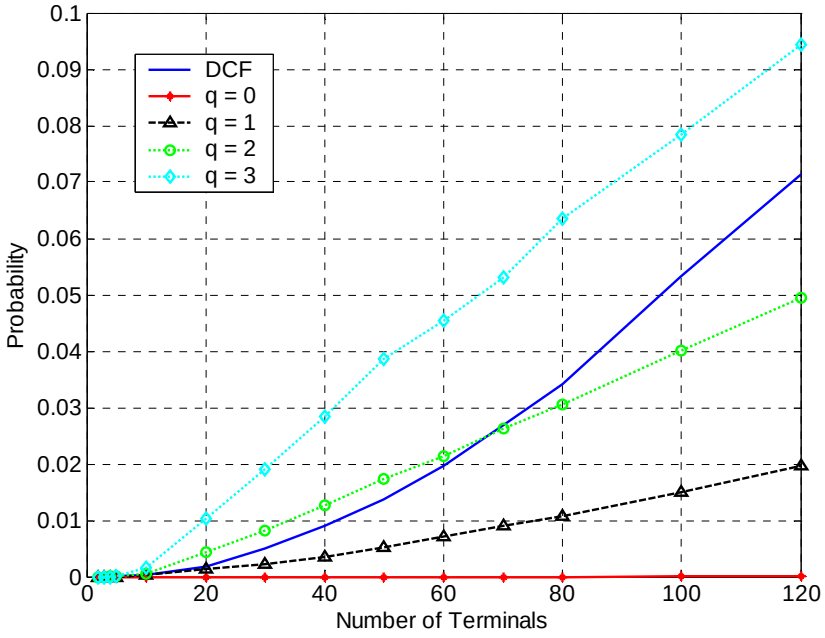

Figure 12: Probability of frame drop (DCF and *q* algorithm)

Figure 12 shows the probability of frame drop for DCF and the *q* algorithm. A frame is dropped if it is retransmitted several times such that the retry limit is reached. Usually the upper layer deals with such cases. As we can see from Figure 12, the probability of frame drop of the *q* algorithm with $q = 0$ is almost zero. That is, nearly all frames reach the destination (some retransmissions may occur). For the *q* algorithm with $q = 1$, the increase in the probability is linear as seen in the figure. Two percent of the frames are dropped when $n = 120$, which is the maximum reached. For $q = 2$, the increase

is also linear but it is higher than the previous case. For DCF, the increase in the number of dropped frames is exponential, and it reaches very high rates when the number of stations, n, increases. For the q algorithm with $q = 3$, the percentage of dropped frames is high. This causes the network to operate in an unstable condition, since the upper layers will continuously be retransmitting dropped frames.

We would expect to have better fairness for the q algorithm compared to the original 802.11 DCF. This is because the q algorithm minimizes collisions and alleviates the effect of a collision (in terms of fairness) if it happens. That is, if a collision occurs between two stations, then the CW will not be doubled immediately. If the CW is to be doubled (as in DCF), then the colliding station will loose its chance in transmitting in the specified interval. Figure 13 shows the fairness for DCF and the q algorithm for $q = 0$, 1 and 2 in the case of two stations. Figure 14 and Figure 15 show the fairness for $n = 5$ and $n = 10$ respectively. We used the sliding window method to calculate the fairness in these figures [23]. The Jain fairness index is used within this method as an index for measuring the fairness of the algorithms (see Section 3.2). We notice that the fairness for the q algorithm and DCF are similar if $n = 2$. However, if $n = 5$ or 10 the fairness of the q algorithm outperforms that of DCF especially for $q = 0$. Notice that if $q = 0$, then the fairness is maximum. This is because the collisions are minimal and each station will get its portion of the overall bandwidth. When $q = 2$, the fairness is better than the case of $q = 1$. The main reason behind this is the mechanics of the q algorithm. That is, if $q = 2$, then the CW will not be doubled until after two collisions. This affects the fairness as explained above.

From the figures, we see that the short-term fairness achieved by the *q* algorithm is better than that of DCF (short-term fairness is mainly the fairness observed over short time periods corresponding to transmitting few frames - see Section 3.2). Consider Figure 14 where $n = 5$. The 0.95 threshold of the fairness is achieved with a normalized window size of 6 for the *q* algorithm if $q = 0$ while it is achieved with a normalized window size of 27 for DCF. This means that in a window of 30 successful transmissions (actual window of frames = Normalized window size * Number of stations), the fairness of the *q* algorithm is about 0.95. In the same range of transmissions (30 successful transmissions), the fairness of DCF is about 0.85, and it will not reach 0.95 unless we have a window of 135 successful transmissions. For $n = 10$ (Figure 15), the difference in favor of the *q* algorithm is even larger. The 0.95 threshold is achieved with a normalized window of size 7 for the *q* algorithm (with $q = 0$). For DCF, the 0.95 threshold is not even achieved in a normalized window of size 50 (corresponding to 500 successful transmissions).

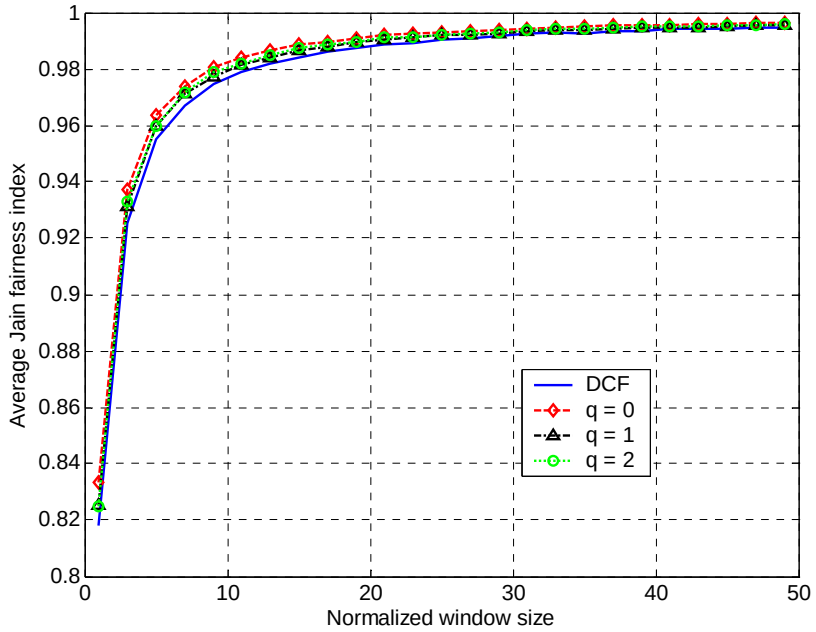

Figure 13: Fairness vs. normalized window size ($n = 2$).

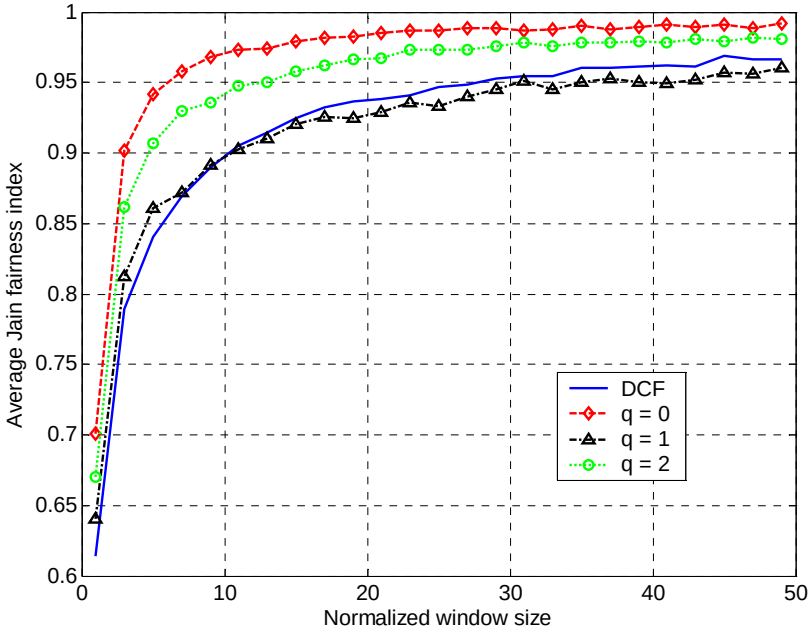

Figure 14: Fairness vs. normalized window size (n = 5).

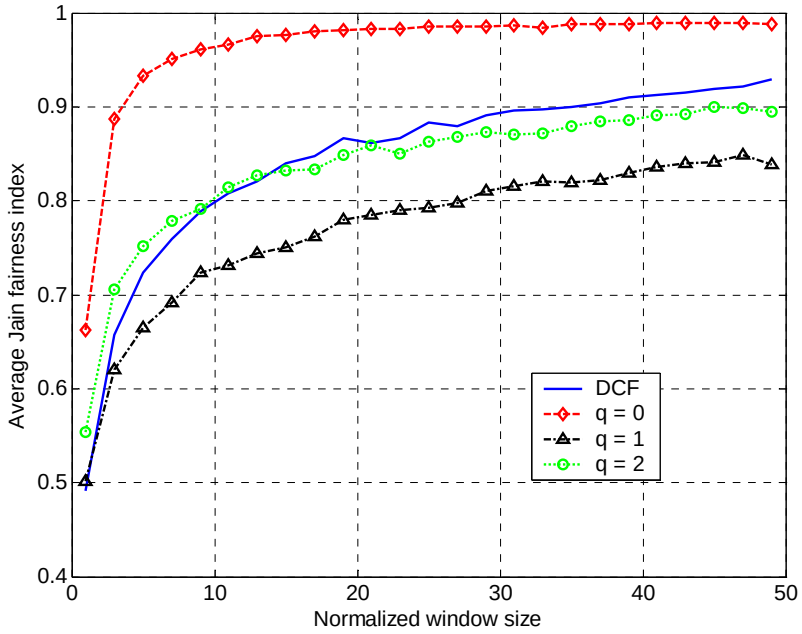

Figure 15: Fairness vs. normalized window size (n = 10).

Similar to the original IEEE 802.11 DCF (Figure 16), the size of the frame has an effect on the performance of the q algorithm. This is shown in Figure 17, Figure 18 and Figure 19 for $q = 0$, $q = 1$ and $q = 2$ respectively. The throughput increases when the packet size increases. Also, the throughput of the 802.11 DCF and the q algorithm with $q = 1$ and $q = 2$ have the same trend. However, when $q = 0$, the performance is completely different. In this case, the throughput increases when n (the number of stations) increases, opposite to the other two cases of the q algorithm. This is the same conclusion we obtained before (see Figure 10). Moreover, in the former two cases, the throughput saturates at 0.85; however, in the case where $q = 0$, the throughput reaches 0.9 with large n and large frame size. This is because the collisions are minimal and successful frame transmissions dominate. Also, the effect of empty slots is minor since the frame is large.

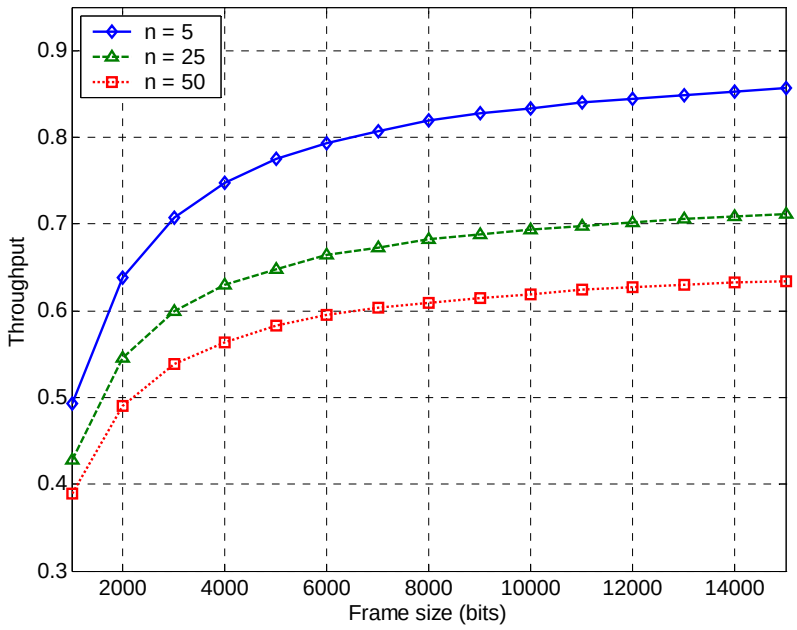

Figure 16: Saturation throughput vs. frame size (802.11 DCF).

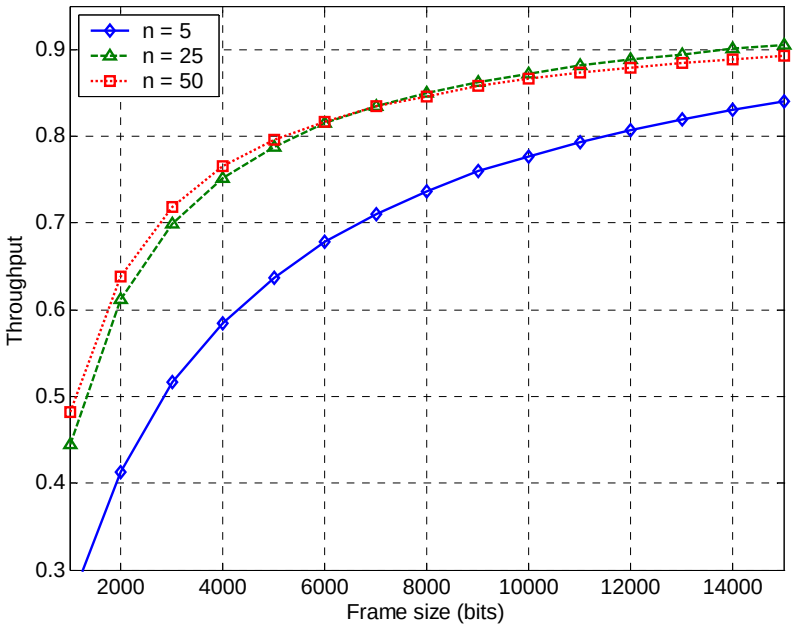

Figure 17: Saturation throughput vs. frame size (q algorithm, $q = 0$).

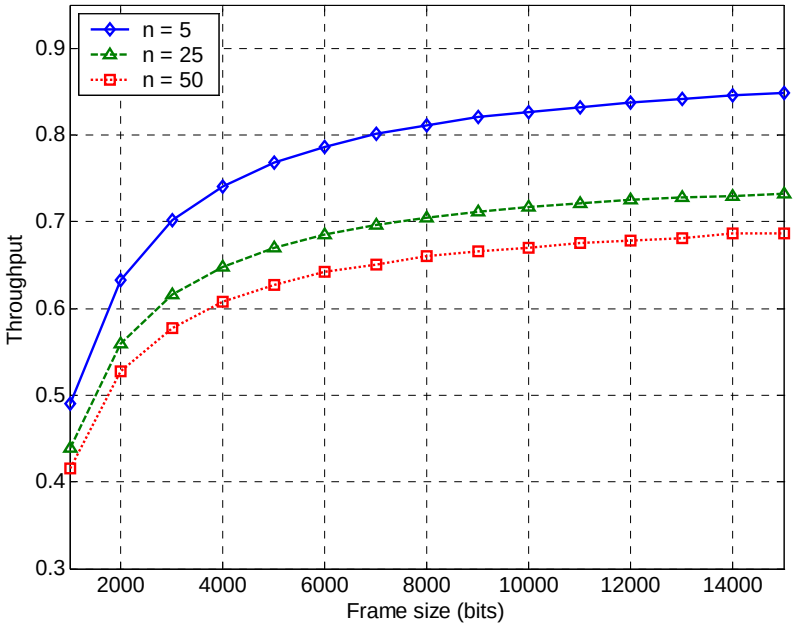

Figure 18: Saturation throughput vs. frame size (q algorithm, $q = 1$).

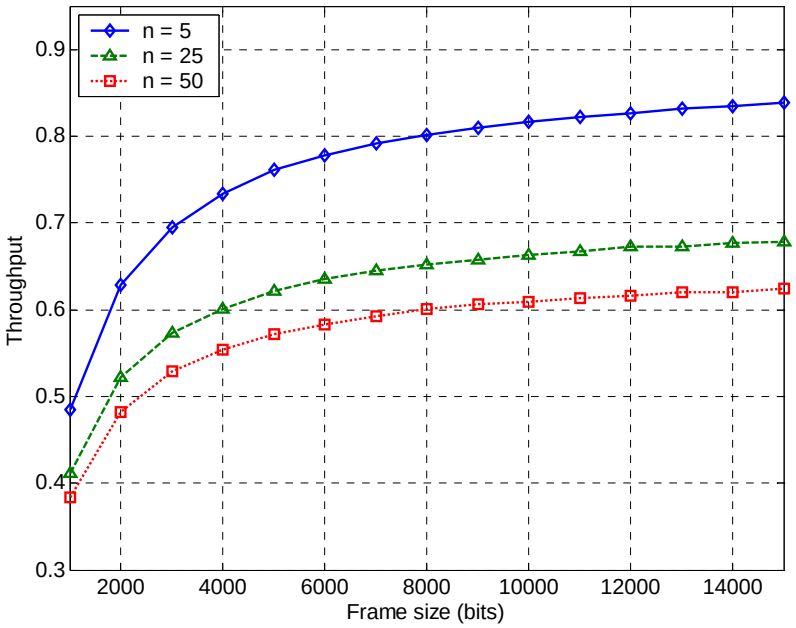

Figure 19: Saturation throughput vs. frame size (*q* algorithm, *q* = 2).

4.2.2 *q* Algorithm at the Non-Saturation State

In this section, we show some of our findings when using the second kind of traffic; that is, the non-saturation state. In this case, we gradually increase the load by decreasing the inter-arrival time. We use a Poisson arrival process simulating the packet arrivals from upper layers. We notice that the network reaches a state where the queue is always full. This is analogous to the saturation state that we implemented in the previous section.

We start by evaluating the DCF function in the non-saturation state. Figure 20 shows the throughput versus the load of the network for the DCF for *n* = 5, 10, 30, 50 and 80. The throughput increases linearly with the load (the x-axis is log scaled). There is a point where the throughput saturates for each number of stations. For example, it saturates at 0.58 when *n* = 80 but at 0.83 when *n* = 5. These values correspond to the values found

in the case of saturation shown in Figure 10. The load at which the network reaches this saturation point differs according to the number of stations. Namely, as the number of stations increases, this load corresponding to the saturation point decreases, and vice versa. That is, fewer stations can accommodate higher load. For example, if $n = 5$, the network saturates when the load per terminal reaches 20 frames/second. However, it will saturate when the load is only 3 frames/second if $n = 30$.

In Figure 21 we see the effect of changing the load on access delay for DCF. As can be seen from the figure, the delay is low when the load is low. However it exponentially increases when the load increases. This indicates that there should be a strict limit when trying to deploy applications (especially real-time) regarding the number of stations and the load put into the network. This is important since a small increase in one of these parameters may cause the network to saturate and operate in an unstable region, and consequently the QoS guarantees may be missed. From the figure, we can see that the average delay of the overall network saturates at some point as the load increases. This saturation point increases as the number of stations in the network, n, decrease; i.e. fewer stations can accommodate higher load. For example, a network consisting of 80 stations will saturate (in terms of delay) when the load per terminal reaches around 1 frame/sec. However, a network consisting of 10 stations will not saturate until the load per terminal reaches 10 frames/second.

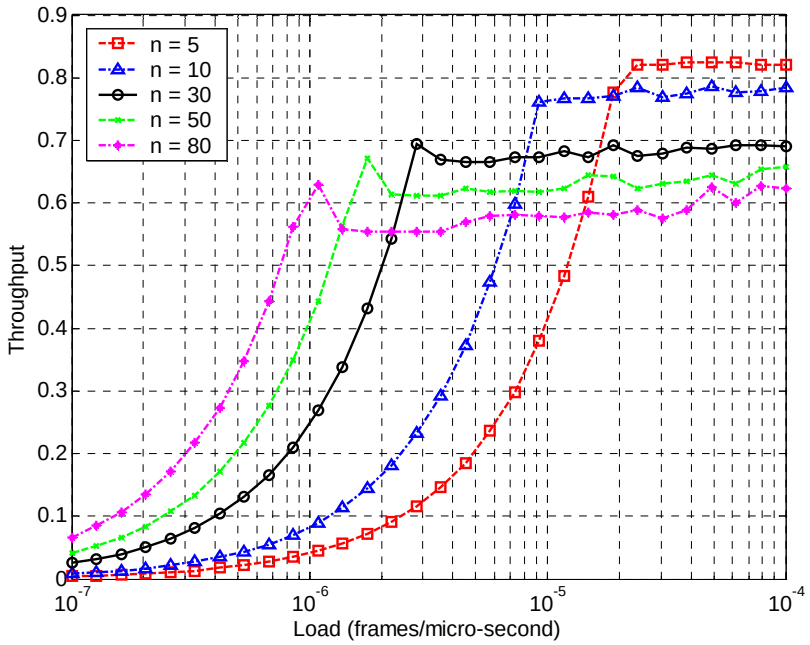

Figure 20: Throughput vs. load per station (802.11 DCF)

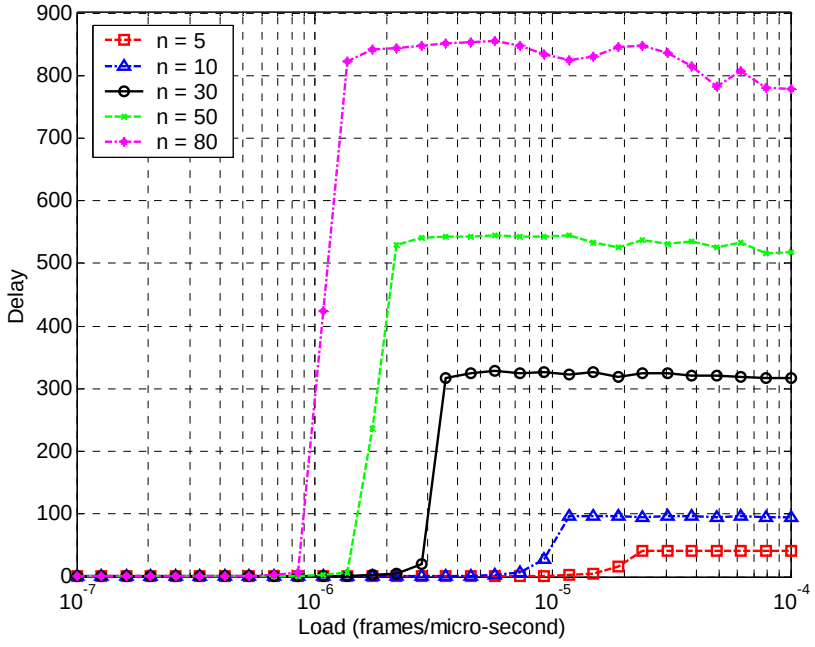

Figure 21: Access delay vs. load per station (802.11 DCF)

In the non-saturation state, the queuing delay can be measured. Figure 22 shows the queuing delay in DCF in non-saturation situations. Similar to the access delay, the queuing delay saturates at some point due to excess load. This saturation point differs depending on the number of stations in the network and the load delivered to the network by each station. This is analogous to the access delay case (Figure 21). One main difference between the two cases is the increasing factor. While the access delay increases exponentially following the saturation point, we see that the queuing delay increases at a slower rate. This rate differs also according to the number of stations. For example, when the number of stations is 80 (n = 80), the queuing delay increases at a fast rate. However, in the case of only five stations (n = 5), the queuing delay increases at a relatively slower rate.

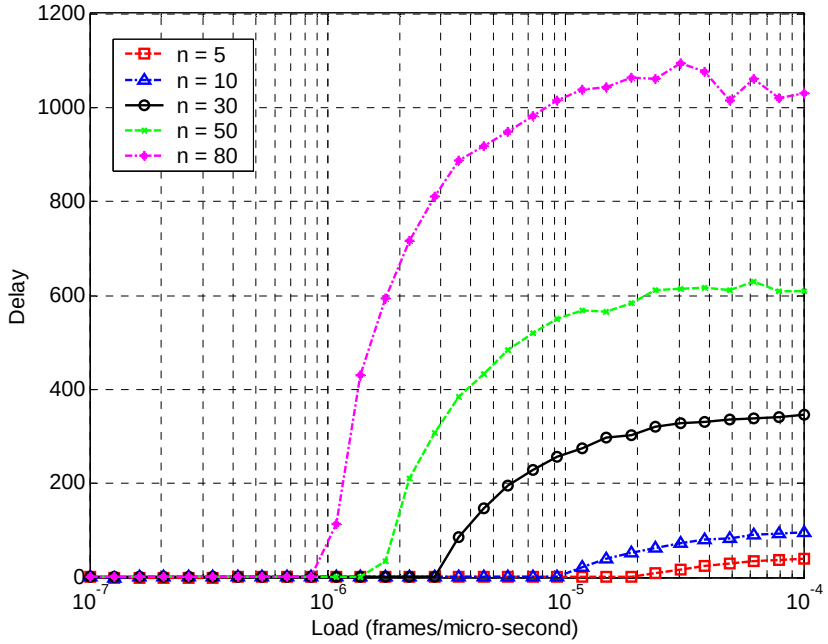

Figure 22: Queuing delay vs. load per station (802.11 DCF)

Figure 23, Figure 24 and Figure 25 show the non-saturation throughput of the q algorithm for $q = 0$, $q = 1$ and $q = 2$ respectively. As can be seen from these figures, the trend of the throughput is the same as in DCF (Figure 20). Among these three cases, larger gain in throughput (compared to DCF) is achieved if $q = 0$. For example, if $n = 30$, the throughput in DCF saturates at 0.7 when the load is 3 frames/second (Figure 20) whereas it saturates in the new algorithm at 0.82 when the load is 3.5 frames/second (Figure 23). This means that the q algorithm with $q = 0$ accommodates more load and delivers more throughput for the same number of stations. If $q = 1$, the saturation throughput gain compared to DCF is larger than that of non-saturation (see Figure 10). That is, if the network is saturated, the difference in throughput between the DCF and the q algorithm with $q = 1$ is larger than the difference between the two algorithms when the network is not saturated (see Figure 24 and Figure 20). Consider the saturation case; the DCF achieves a throughput of 0.55 when $n = 80$ while the q algorithm with $q = 1$ achieves 0.63 with the same number of stations. In the non-saturation case, the two algorithms achieve a throughput of 0.55 and 0.6 respectively. The difference in the first case is 0.08 while it is 0.05 in the second case.

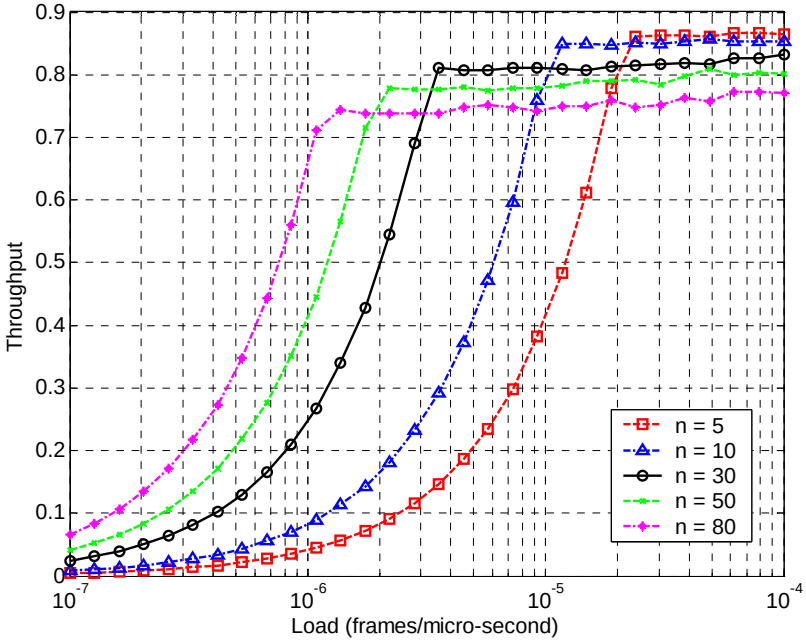

Figure 23: Throughput vs. load per station (q algorithm, $q = 0$)

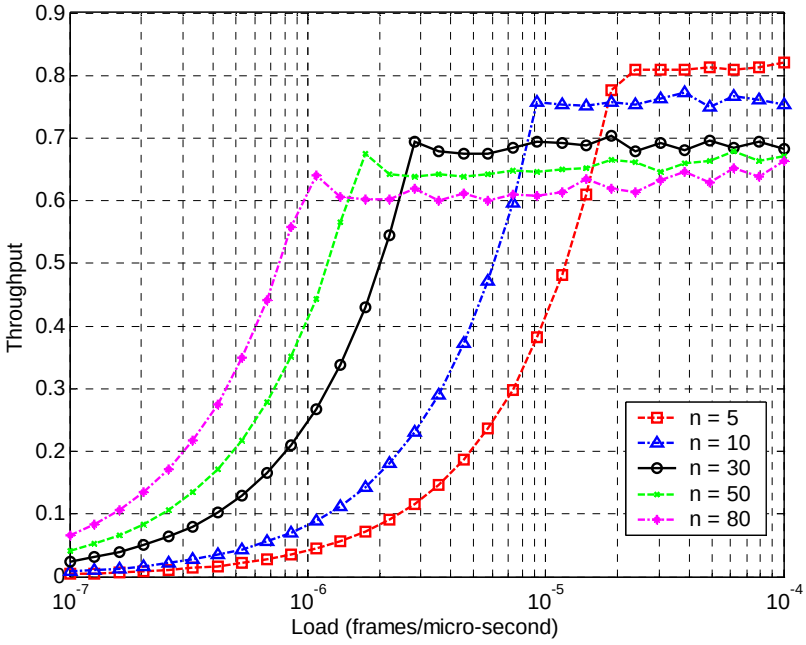

Figure 24: Throughput vs. load per station (q algorithm, $q = 1$)

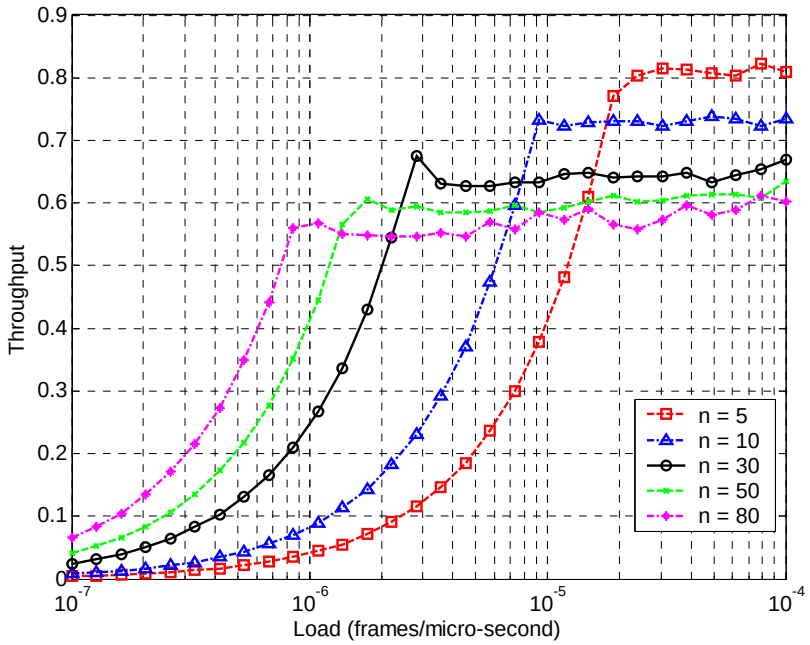

Figure 25: Throughput vs. load per station (q algorithm, $q = 2$)

The access delay of the new algorithm for $q = 0$, $q = 1$ and $q = 2$ in the case of non-saturation is shown in Figure 26, Figure 27 and Figure 28 respectively. Here, the trend is the same as in DCF. The saturation point in terms of load has also the same trend. As in the saturation case, a better delay (the lower delay) can be achieved if $q = 0$.

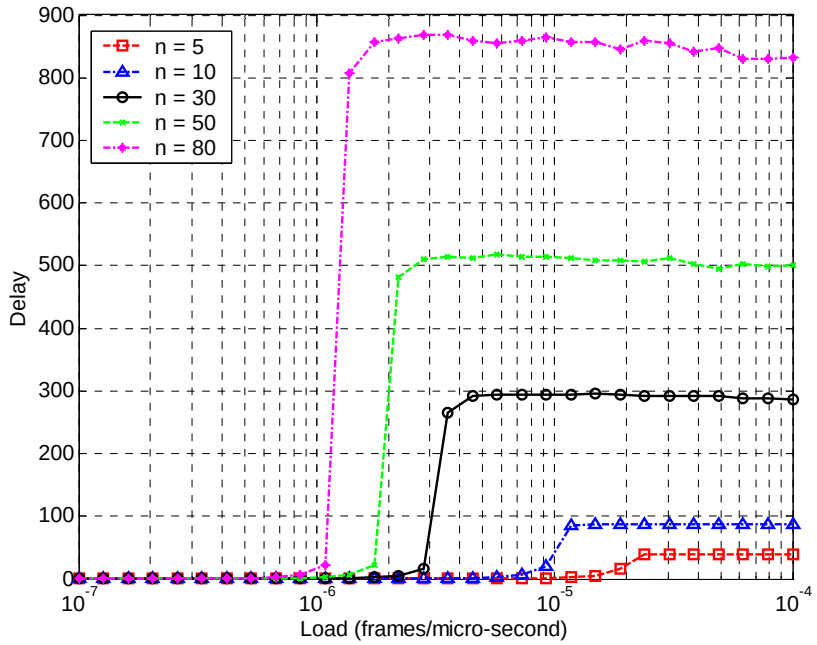

Figure 26: Access delay vs. load per station (q algorithm, $q = 0$)

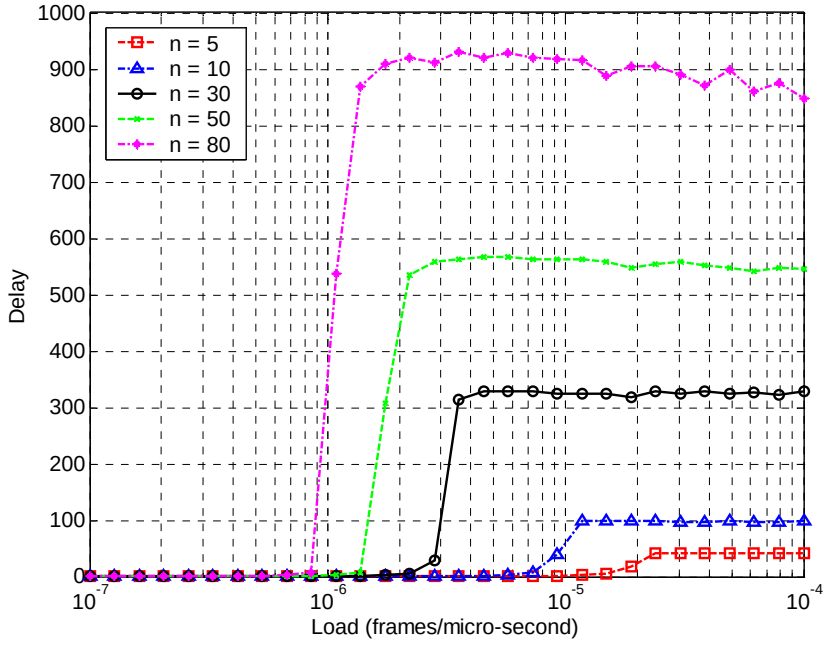

Figure 27: Access delay vs. load per station (q algorithm, $q = 1$)

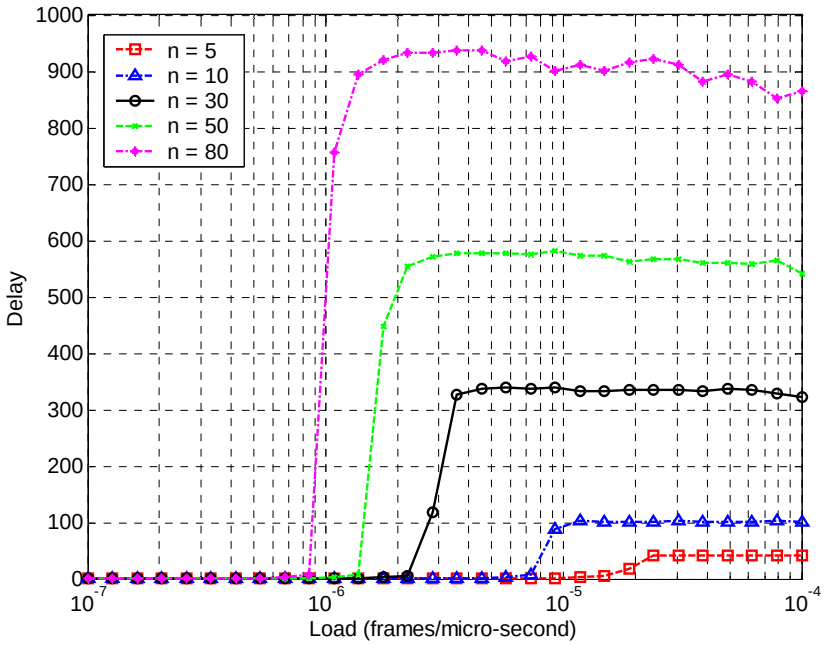

Figure 28: Access delay vs. load per station (q algorithm, $q = 2$)

Figure 29, Figure 30 and Figure 31 show the queuing delay of the new algorithm for $q = 0$, $q = 1$ and $q = 2$ respectively. From the figures, we see that the queuing delay increases slower than the access delay. For the queuing delay, there are larger difference between DCF and the new algorithm when $q = 0$. For example, for $n = 80$ and a load of 10 frames/sec, the delay of the new algorithm is 800 msec (Figure 29) whereas it is 1000 msec for DCF with the same load and number of stations (Figure 22). When $q = 1$ or 2, the difference is negligible.

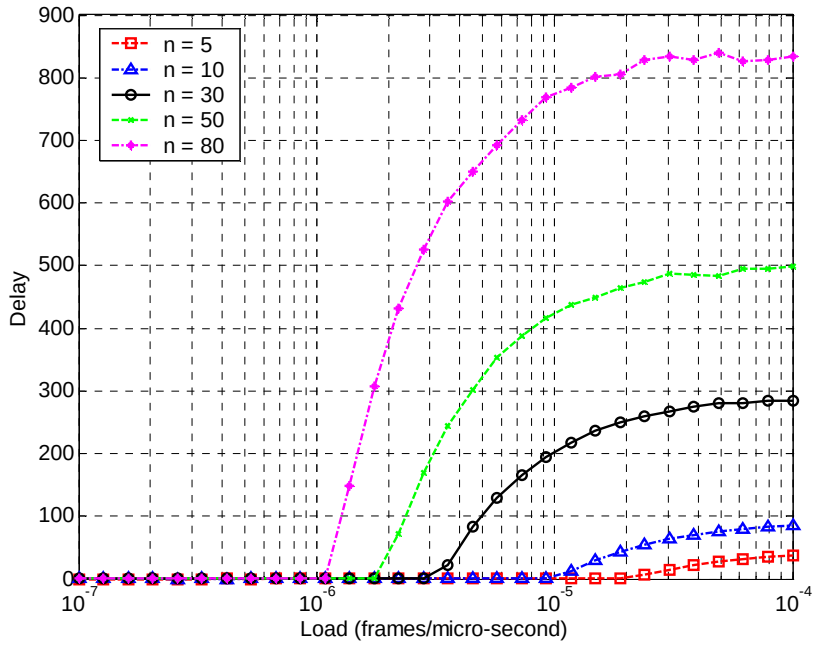

Figure 29: Queuing delay vs. load per station (q algorithm, $q = 0$)

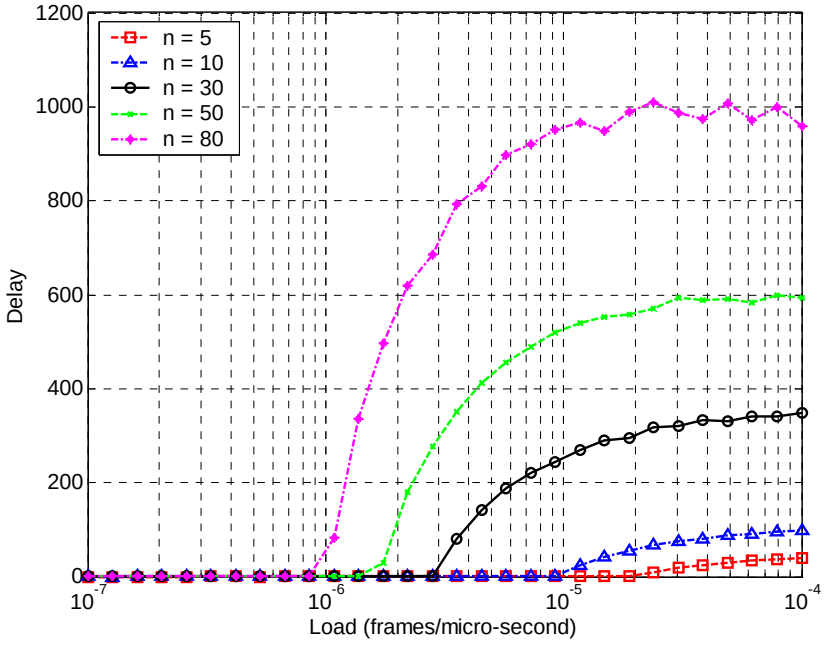

Figure 30: Queuing delay vs. load per station (q algorithm, $q = 1$)

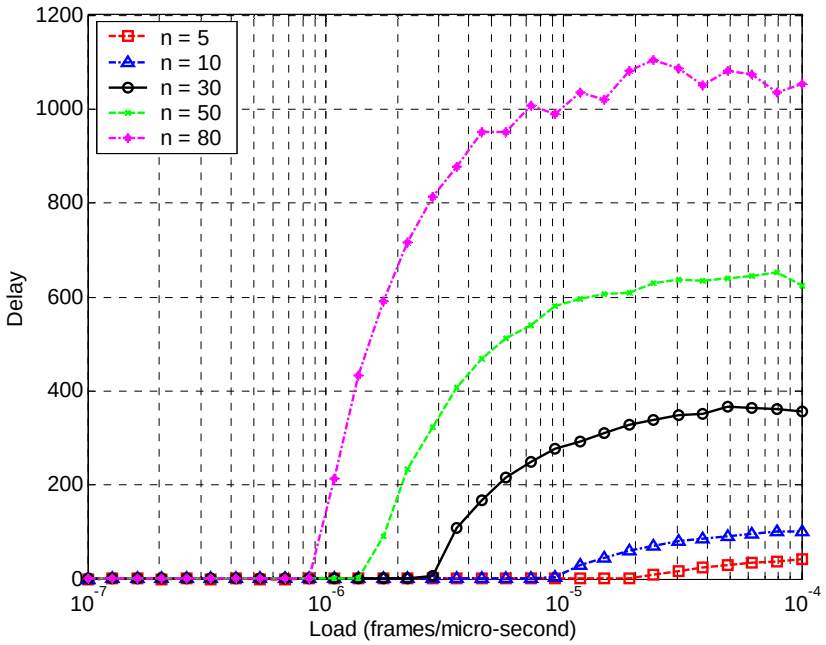

Figure 31: Queuing delay vs. load per station (q algorithm, $q = 2$)

4.3 Differentiation in the *q* Algorithm to Support QoS

One of the main goals for introducing the q algorithm is to achieve QoS support through differentiation. The q algorithm is a good candidate for this purpose. As shown previously in Sections 4.2.1 and 4.2.2, the performance of the protocol depends strongly on the choice of the parameter q. By assigning each station a different value for q, we could achieve differentiation.

One scenario is to differentiate according to the throughput needed for each flow type. For example, we may assume the number of stations is high and we may assume three types of applications or data flows; say file download, Internet browsing and email. In the file download application, the q parameter will obviously be set to zero. This option

gives the highest throughput when we have large number of stations (see Figure 10). Of course high throughput is needed for file download. The Internet browsing would be assigned a value of one for the q parameter. Moreover, the download stream can be assigned a value of zero while the upload stream remains one. This is logical since the upload does not carry much traffic while the download does. Finally, we can assign the q parameter of the email application a value of two or maybe three. The email usually does not carry much traffic, and the time limit is not strict.

As another example, consider the following traffic flow types: video conferencing, voice call and instant messaging. The straightforward assignment for the q parameter of the three application types would be respectively zero, one and two for the video conferencing, the voice call and the instant messaging. Video conferencing carries much data, and it has fixed time limits. Therefore, it is logically assigned zero for the q parameter. The voice call carries less data but also has fixed time constraints, and so it can suitably be assigned one for its q parameter. This would provide high throughput (although not as high as the case where $q = 0$) while maintaining low latency (see Figure 11). Finally, the instant messaging application does not require much throughput of the channel to be reserved, but it requires immediate transmission of the data. Hence, a value of two or maybe three for the q parameter is appropriate for such type of application. This is clear from Figure 10 and Figure 11 where we can see that the throughput of the network is lower when $q = 2$ but the average access delay is comparable to the delay when $q = 1$.

The above examples apply if the number of stations is large. However, if we have fewer stations ($n < 10$), then the opposite is the solution for such applications. If the number of stations is not known beforehand, the differentiation is still valid. However,

different applications may not receive the expected throughput share, and they may not experience the appropriate delay.

The above is a theoretical reasoning that depends on the performance figures of the q protocol and the different application needs. Simulation studies need to be performed to gain confidence in the above expected results. These simulations should take into account different network situations and different implementations of the applications already discussed (and maybe others).

We depend on the results of the saturation state in our theoretical analysis of the differentiation mechanisms in the q algorithm. In the saturation conditions, the performance figures obtained indicate the worst case scenario. This is more appropriate since some applications generate large amounts of data to be delivered through the wireless network. It is not appropriate to discuss the differentiation mechanisms relying on the non-saturation performance figures. This is because we should expect worst case scenarios, and this is done by anticipating large amount of traffic; if we do not do so, some scenarios may occur unexpectedly.

CHAPTER 5

THE TWO-STAGE ALGORITHM

5.1 Fixed CW Attempt

From Figure 10 we see that the throughput when $q = 0$ is maximized if $n > 8$. As explained in Section 4.2.1, this is a special case in the q algorithm. The CW in this case is never set back to its minimum; it is fixed at the maximum. This leads us to the idea of fixing the CW to a specific value so that the collisions are minimized and the throughput is maximized, as in Figure 10 for $q = 0$. Figure 32 shows the throughput versus the number of stations in a saturation state for three different sizes of the CW (fixed). We notice that the behavior is similar to the q algorithm with $q = 0$. This is expected since the latter behaves as if the CW is fixed at the maximum size, 1024 in this case. It is clear that the new idea (fixed CW) gives a poor performance if the number of stations, n, is low. This is because the time is wasted with empty slots. However, when n is relatively large (> 10), the performance is better than DCF. This is because the number of collisions is reduced when the CW is large, even for a large number of stations.

As can be seen from Figure 32, a larger CW will be suitable for a larger number of stations. On the other hand, if we choose a small CW size, this would produce good results for a low number of stations at the expense of reduced throughput for a large

number of stations. This leads us to the next step in our attempt to improve the DCF function; the two-stage algorithm.

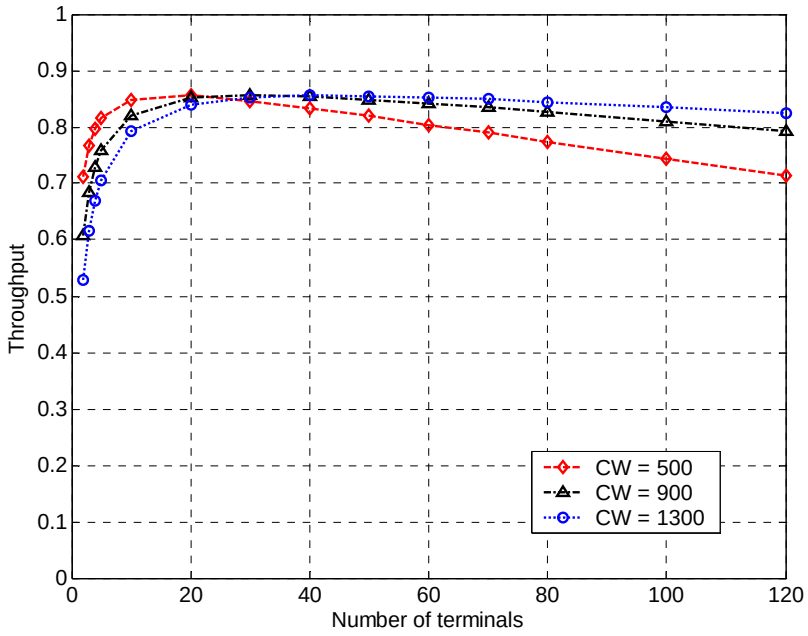

Figure 32: Saturation throughput vs. number of terminals (CW fixed)

5.2 Description of the Two-Stage Algorithm

Following the discussion in Section 5.1, we propose a new and simple algorithm to enhance the DCF function. In this new algorithm (called Two-Stage Algorithm), we try to find a solution for performance degradation in high loaded situations (large n) without causing harmful consequences in other situations. That is, we want to avoid the disadvantages of the q algorithms with $q = 0$ (4.2.1) and the fixed CW algorithm (5.1). In these two algorithms, the throughput is low if $n < 10$. This is because the CW is set to a large value and the time is wasted with empty slots. As a solution to this problem, we

define a smaller set of CW sizes consisting of two stages only. The sizes of the CW in these two stages should be chosen according to a strategy that maximizes the throughput in both low loaded and high loaded situations. A suitable choice is to have the CW with a small size in the first stage and a large size in the second stage. We call the CW at the first stage CW_{min} and the CW at the second stage CW_{max}.

The two-stage algorithm is simple, and it works as follows. The CW is initially set to CW_{min}. If a collision occurs, the CW is increased and set to CW_{max}. If another collision occurs, no further increase in the size of the CW is done; i.e. the CW is fixed at CW_{max}. If a successful transmission occurs at any time, the CW is minimized and set to its minimum value CW_{min}. In brief, the two-stage algorithm works the same as DCF but with two stages only.

5.3 Markovian Analysis

In this section, we introduce an analytical model of the two-stage algorithm in saturation conditions. In our model, we assume that at any time each station has a frame ready to be transmitted; i.e. the buffer of the station is never empty. Relying on the assumption that the state of a station is independent of that of the other stations (as in [6] and [22]), our model represents the behavior of a single station. The model is shown in Figure 33. We follow the model proposed in [22] since it is simpler and more compact.

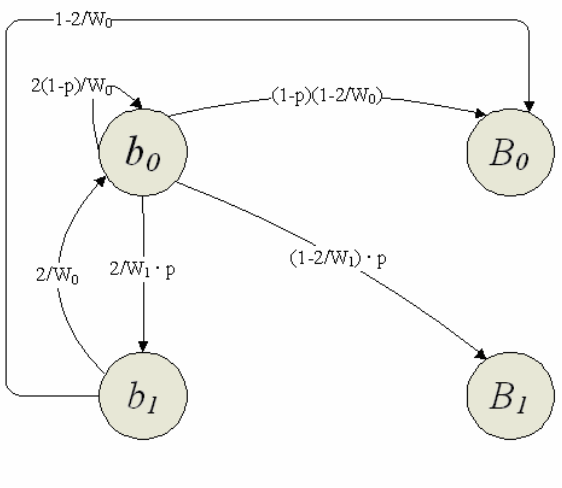

Figure 1: Markov chain of the two-stage algorithm

The states labeled with b_i represent the station with backoff counter equal to zero; i.e. the case where the station actually transmits a frame in the current step. States labeled with B_i represent the station while it is decrementing its backoff counter (and it did not reach zero). The states have an index of 0 or 1 representing the backoff stage, where 0 is the first stage ($CW = CW_{min}$), and 1 is the second stage ($CW = CW_{max}$). We denote by W_i the contention window size at backoff stage i; in our case, $W_0 = CW_{min}$ and $W_1 = CW_{max}$. Finally, p in the model represents the collision probability seen by a station transmitting on the channel.

In our research, we depend on simulations for finding results of the two-stage algorithm. Future research will evaluate the analytical model and get a final analytical result, as well as comparing it with the simulation results.

5.4 Results

Figure 34 shows the throughput of the new algorithm with different values of CW_{min} and CW_{max}. Also shown in the figure is the throughput of DCF and the *q* algorithm with $q = 0$ for comparison.

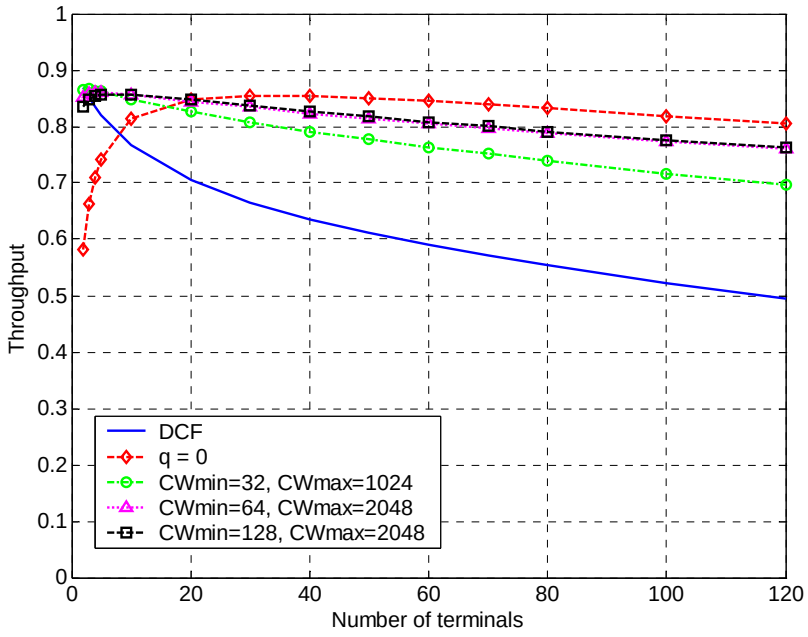

Figure 34: Saturation throughput vs. number of terminals (two-stage algorithm)

As can be seen in the figure, the two-stage algorithm achieves the best performance among other algorithms on average. If $CW_{min} = 32$ and $CW_{max} = 1024$, the performance is better than DCF for all *n*. It is also better than the *q* algorithm when $n < 16$. When $n = 120$ the difference in throughput between the two algorithms (two-stage algorithm and the *q* algorithm) is only 0.1; otherwise, it is less than that. When $CW_{min} = 64$ or 128 and $CW_{max} = 2048$, the two-stage algorithm is also better than DCF for all *n* (except for $n = 2$ and $n = 3$ with very small difference). Figure 34 also shows that the two-stage algorithm, with

these parameters, is better than the q algorithm if $n < 20$. When $n = 120$, the difference in throughput between the two algorithms is only 0.05. This is the maximum difference in throughput. In other situations, the difference is less than that.

When comparing the different algorithms in this section, we should take into account their overall performance in all situations; that is, the performance in both the high loaded situations (large number of stations) and the light loaded situations (few stations). The DCF performs well in light loaded situations, but its performance degrades in high loaded situations. This was explained in detail in Section 2.2.1. In brief, the CW is immediately decreased to its minimum following a successful transmission. This leads to a collision with a high probability when the network is congested. Also, multiple collisions occur before the CW reaches a large value (a value suitable for large n). On the other hand, the q algorithm performs better in high loaded situations while its performance is poor when the number of stations is low. This was also explained in Section 4.2.1. Briefly, the throughput is wasted by empty slots when the number of stations is low and eventually the throughput is degraded. The q algorithm performs better in high loaded situation since the CW is sufficiently large and collisions are minimized.

The two-stage algorithm has the advantage of the two algorithms, and hence it solves the problem of throughput degradation in both situations. In the light loaded situations, the number of collisions is small, and the CW is set to CW_{min} most of the time. This leads to an improved throughput compared to the q algorithm. In the high loaded situations, the two-stage algorithm performs better than the DCF. The DCF deals with collisions as follows. The backoff procedure is invoked and the CW is increased exponentially with each collision. Until the CW reaches the suitable size where collisions

are minimized, multiple collisions occur causing the throughput to degrade. On the other hand, the two-stage algorithm deals differently with collisions. When a collision occurs, the CW is immediately set to CW_{max}. This leads to a recent successful transmission in most cases.

The throughput of the two-stage algorithm is less than that of the *q* algorithm when the network is highly loaded. The reasoning behind this small decrease is as follows. The *q* algorithm behaves as if the size of CW is fixed. This means that the collisions are rare and hence the throughput is maximized. On the other hand, in the two-stage algorithm, the CW oscillates between two values: CW_{min} and CW_{max}. This means that in highly loaded situations there will be collisions, and these collisions occur most probably when the CW is set to CW_{min} (when the frame is sent for the first time). Actually, even if the CW_{min} is enlarged more, there will still be collisions, and hence there will still be degradation in the throughput.

To clarify this result, we implement the two-stage algorithm with different values for CW_{min} while fixing the size of CW_{max}. Figure 35 shows the throughput for different cases of CW_{min}. It is clear from the figure that increasing the size of CW_{min} does not help in improving the throughput of the network in highly loaded situations. On the other hand, enlarging the size of CW_{max} will improve the performance of the algorithm (in terms of throughput) in highly loaded situations. Figure 36 clearly shows this. Namely, when $CW_{max} = 512$, the throughput is 0.6 when the number of stations, *n*, is 120. The throughput improves if CW_{max} is chosen to be 1024 and even improves further when $CW_{max} = 2048$ (see Figure 36).

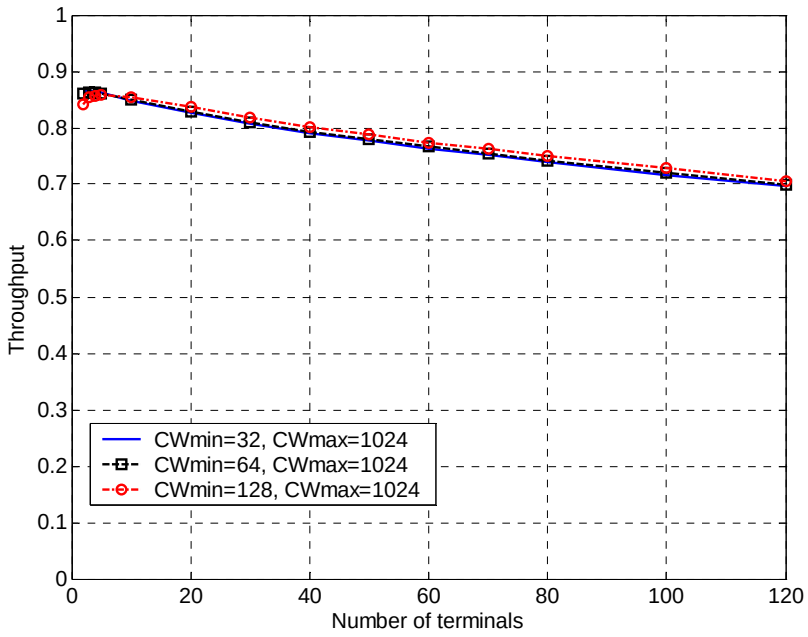

Figure 35: Saturation throughput vs. number of terminals (two-stage algorithm)

One would expect that throughput of the two-stage algorithm will degrade in light loaded situations if we choose a relatively large value for CW_{min}. Figure 35 shows that this is not totally true (the difference in throughput in this case is negligible). This leads to the conclusion that collisions are the main factor that causes the throughput to degrade, and not the empty slots due to relatively large sizes of the CW. One exception to this happens if we choose CW_{min} to be very large (say 512 or 1024). In this case there will be degradation. Moreover, if we choose CW_{min} to have the same value as CW_{max}, we end up implementing the fixed CW algorithm described in Section 5.1.

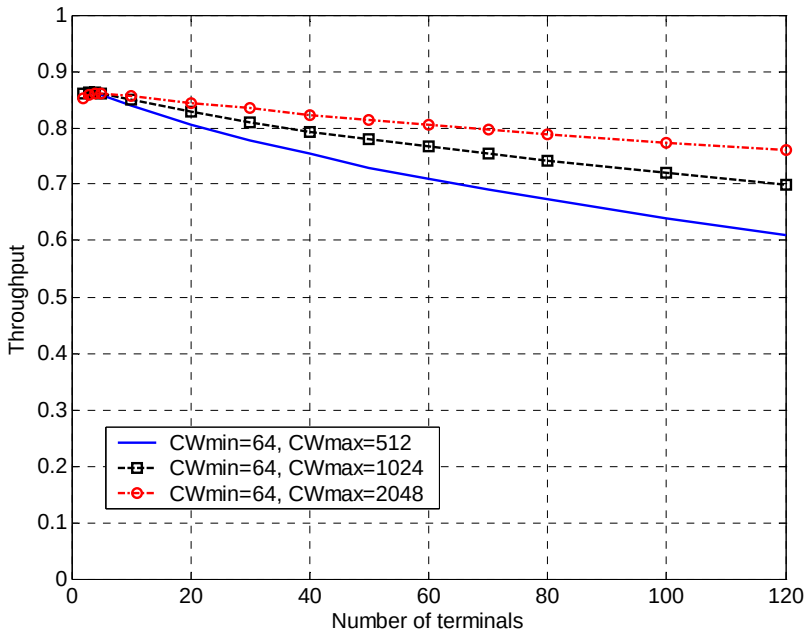

Figure 36: Saturation throughput vs. number of terminals (two-stage algorithm)

It is interesting to compare the throughput of the two-stage algorithm with a theoretical limit of the throughput described in [14]. This theoretical limit is calculated therein by defining an analytical model called the "*p*-persistent IEEE 802.11 protocol". "The *p*-persistent IEEE 802.11 protocol differs from the standard protocol only in the selection of the backoff interval. Instead of the binary exponential backoff used in the standard, the backoff interval of the *p*-persistent IEEE 802.11 protocol is sampled from a geometric distribution with parameter *p*" [14]. The authors show that the standard IEEE 802.11 DCF protocol with a constant size CW tuned to the optimal *p* value has a capacity close to the theoretical limit. That is, with each *n*, we need to choose a CW with an appropriate (constant) size that maximizes the throughput. In this case, the size of the CW will not be fixed for all *n* as in Section 5.1; instead, for each *n* there is a constant CW size. This means that the *p*-persistent IEEE 802.11 protocol acts like a single-stage algorithm.

This is achieved by dynamically setting the size of the CW depending on the number of stations, n, in the network; i.e. CW size is set at run time.

Figure 37 shows the comparison between the throughput of the two-stage algorithm and the theoretical limit of the throughput defined in [14] along with the throughput of DCF and the q algorithm with $q = 0$. As can be seen from the figure, the q algorithm achieves better throughput than the theoretical limit in some situations. This shows that either the theoretical limit defined in [14] is not accurate or that the difference in simulation parameters (e.g. modulation mode, packet size, bit rate…etc) may influence the results. It is outside the scope of this research to find the theoretical limit of the throughput with specific parameter values or to reproduce other results for this limit calculation.

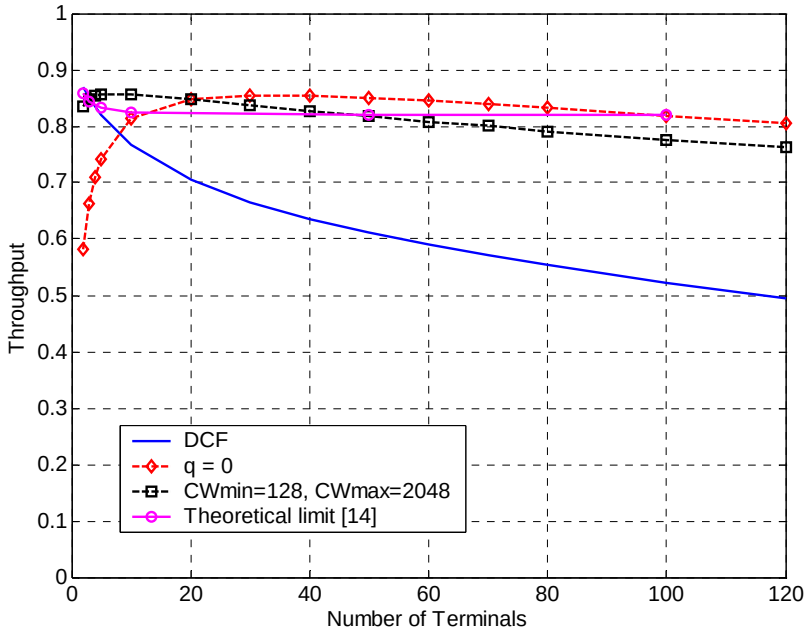

Figure 37: Saturation throughput vs. number of terminals

It is expected that there will be a difference in the throughput between the two-stage algorithm and the theoretical limit, since they function differently. We show the comparison between these in Figure 38 for clarification. It is clear from the figure that the two-stage algorithm can achieve higher throughput than the theoretical limit if n is greater than 3 and less than 50. In other situations (i.e. $n = 2$ or $n > 50$), the throughput of the two-stage algorithm does not reach the theoretical limit. However, the difference is small even if $n = 100$; it is only 0.04. Unfortunately, the data available for the theoretical limit is up to $n = 100$.

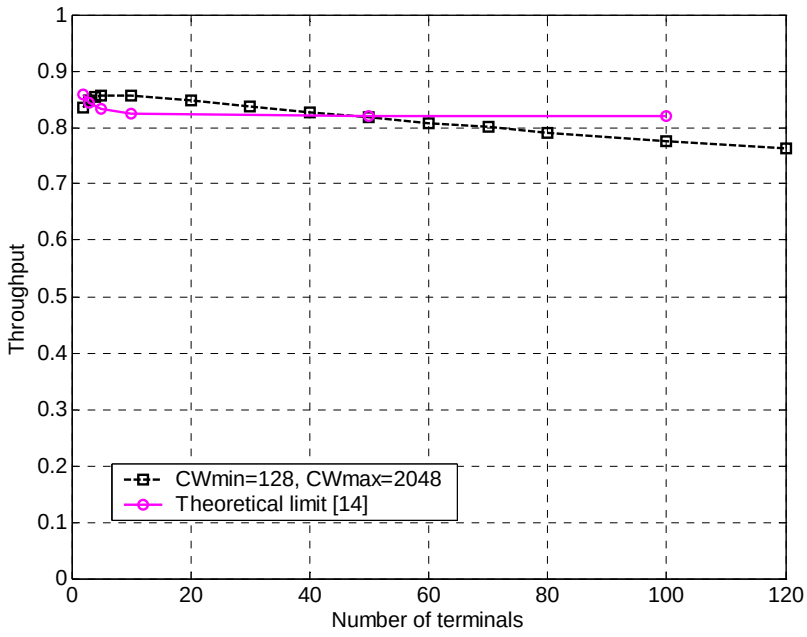

Figure 38: Saturation throughput vs. number of terminals

The practical method for achieving the throughput limit used in [14] is to choose the size of CW dynamically; i.e. at run time. Moreover, the size of CW should be optimized for the number of stations, n. A station could basically estimate the number of stations in the network by (continuously) counting the number collisions, successful transmissions

and empty slots and then making some calculations on these values. This adds to the complexity of the algorithm. On the other hand, the two-stage algorithm achieves comparable (and sometimes better) results with a much simpler procedure. The stations need not do any calculations at run time.

Let us now consider the delay of the two-stage algorithm. We can see from Figure 39 that the q algorithm with $q = 0$ experiences the lowest delay. The CW in the q algorithm is fixed at the maximum, and hence collisions are minimized. The highest delay is experienced by DCF. In DCF, multiple collisions in highly loaded situations cause the delay to increase. As discussed in Section 4.2.1, the performance of DCF when $n > 80$ is not trusted because of high frame drop (see Figure 12). The performance of the two-stage algorithm in terms of delay is good, especially when CW_{min} is large. When CW_{min} gets larger, the number of collisions becomes less. However, the number of collisions is not as low as in the case of q algorithm with $q = 0$.

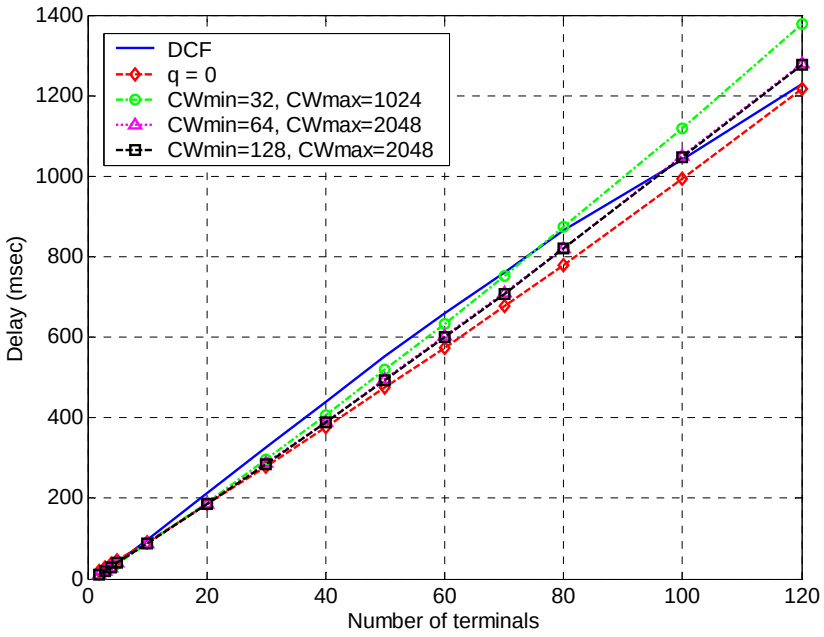

Figure 39: Access delay vs. number of stations (two-stage algorithm)

Now, let us consider the fairness of the two-stage algorithm, which is an important metric. As in the case of throughput, the fairness of the two-stage algorithm depends on the choice of CW_{min} and CW_{max}. Figure 40 shows the fairness for the two-stage algorithm along with the q algorithm with $q = 0$ and the DCF, all for $n = 2$. As can be seen from the figure, the q algorithm with $q = 0$ achieves the highest fairness. In the q algorithm with $q = 0$, the CW is fixed at the maximum and collisions are minimized. Hence, each station will transmit in the shortest possible time. That is, if each station successfully transmits its frames (i.e. without collisions), then each station will receive a fair share of the total bandwidth. However, if a station's transmission faces a collision, then this station will initiate its backoff procedure, causing the transmission to be delayed. Meanwhile, other stations may succeed in transmitting their frames (which may be more than one for each station) causing the fairness to be reduced. Actually, in [24] the research mainly depends

on studying the number of inter-transmissions between two transmissions of the same station. DCF has similar fairness; the difference appears when the number of stations increases (see Figure 13, Figure 14 and Figure 15 in Section 4.2.1). The two-stage algorithm achieves good fairness when $CW_{min} = 128$ and $CW_{max} = 2048$. In this case, the protocol achieves the 0.95 threshold with a normalized window size of 8. For other values of CW_{min} and CW_{max}, the fairness is less, as shown in Figure 40.

Notice the difference between throughput and fairness regarding the behavior of the two-stage algorithm. In Figure 34, we can see that the throughput values of two cases of the two-stage algorithm are similar when $CW_{min} = 64$ or 128 and $CW_{max} = 2048$. However, when examining the fairness for the same values of CW_{min} and CW_{max} (Figure 40), we see that there is a large difference in this case.

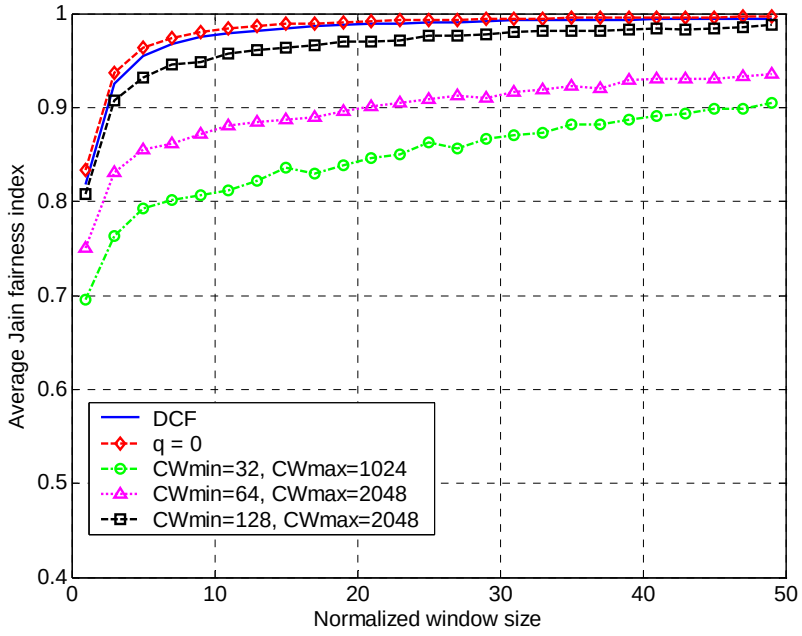

Figure 40: Fairness vs. normalized window size ($n = 2$)

If we set CW_{max} to 1024 and CW_{min} to 64 or 128, we end up with an improved fairness for the two-stage algorithm. This is shown in Figure 41. When $CW_{min} = 128$ and $CW_{max} = 1024$, the fairness is comparable to that of the q algorithm with $q = 0$ and DCF. In this case, the 0.95 threshold is achieved when the normalized window size is only five.

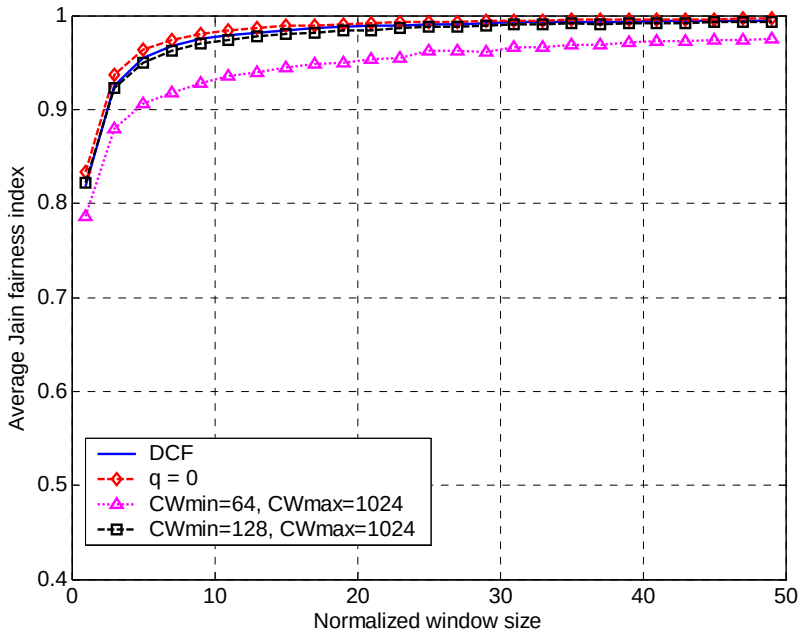

Figure 41: Fairness vs. normalized window size ($n = 2$)

This leads us to investigate the effect of changing the main settings of the two-stage algorithm (i.e. CW_{min} and CW_{max}) on fairness. Figure 42 shows the fairness of the two-stage algorithm with different settings. Here, we fixed CW_{min} at 64 and varied CW_{max} each time. We can notice from the figure that the fairness improves as CW_{max} decreases. This may be caused by the fact that the station will have the chance to transmit earlier in this case (when CW_{max} is smaller). That is, empty slots affect fairness negatively in the case of large CW_{max} by introducing some delay for the transmission of a station; during the period of backoff, other stations may transmit more than one frame, causing the fairness to

degrade. Opposite to this criterion, the throughput of the two-stage algorithm increases when CW_{max} increases. This was shown in Figure 36. One metric may be penalized for the other if needed (the throughput is usually more important). Otherwise, an optimized solution should be found according to the need.

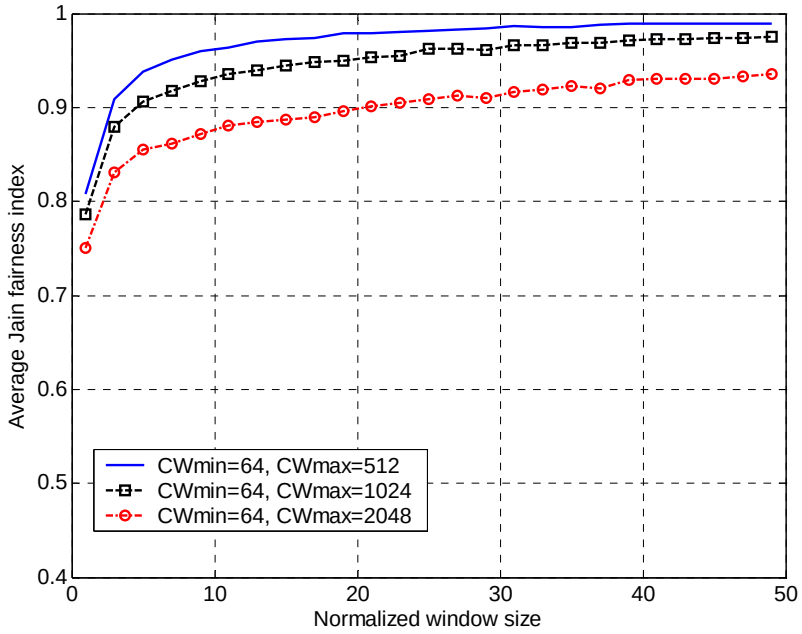

Figure 42: Fairness vs. normalized window size for the two-stage algorithm ($n = 2$)

The same thing can be said about fixing CW_{max} and varying CW_{min}. That is, the throughput and fairness values are again affected differently. While the throughput is not affected by varying CW_{min} as shown in Figure 35, the fairness of the two-stage algorithm is noticeably affected. As shown in Figure 43, the fairness improves as CW_{min} increases. If CW_{min} is small, a frame to be transmitted is subject to collisions. This gives the chance for other stations' frames to be transmitted during the backoff invocation of the frame in hand. This in turn leads to degraded fairness. On the other hand, when CW_{min} is large,

collisions are reduced. Therefore, each station can get its turn in transmitting its frames in (relatively) shorter time and get a fair share of the bandwidth.

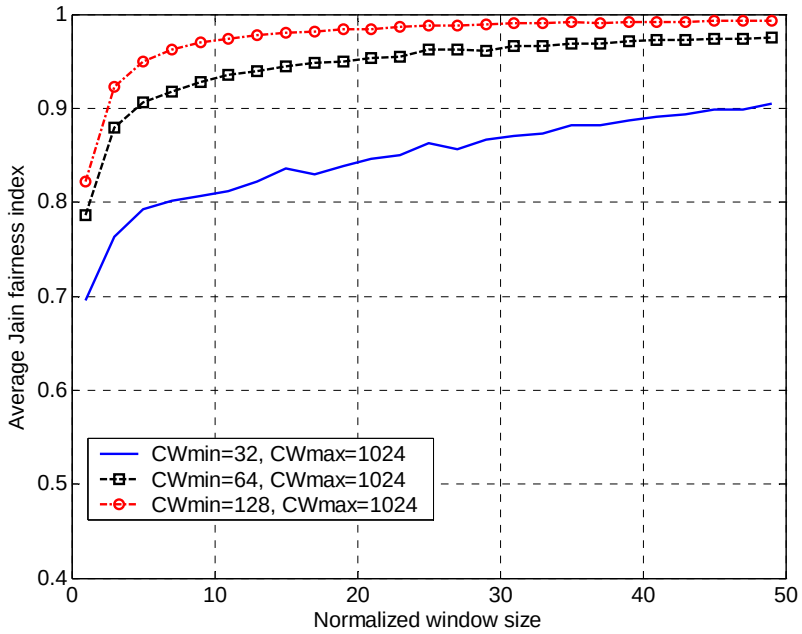

Figure 43: Fairness vs. normalized window size for the two-stage algorithm ($n = 2$)

Let us now consider increasing the number of stations and observing the fairness of the two-stage algorithm. Figure 44 shows the fairness of the q algorithm with $q = 0$, the DCF and the two-stage algorithm, all for $n = 5$. The fairness of the q algorithm with $q = 0$ is the highest. After that comes the fairness of the DCF and the fairness of the two-stage algorithm with $CW_{min} = 128$ and $CW_{max} = 1024$ with a very small difference in between. When $CW_{min} = 64$, the fairness of the two-stage algorithm is degraded as shown in the figure.

Figure 40, Figure 41 and Figure 44 along with Figure 34 show that the throughput of the two-stage algorithm outperforms that of the DCF. However, the fairness of DCF is in general better than that of the two-stage algorithm. The two-stage algorithm helps in

improving the overall throughput of all stations, especially in high loaded situations. Moreover, a good choice of CW_{min} and CW_{max} will produce an acceptable performance in terms of fairness (which is comparable to DCF). On the other hand, the throughput of DCF degrades in high loaded situations. Moreover, the throughput shown in these figures is the maximum throughput (saturation throughput) and it cannot be enhanced without an intervention in the DCF function.

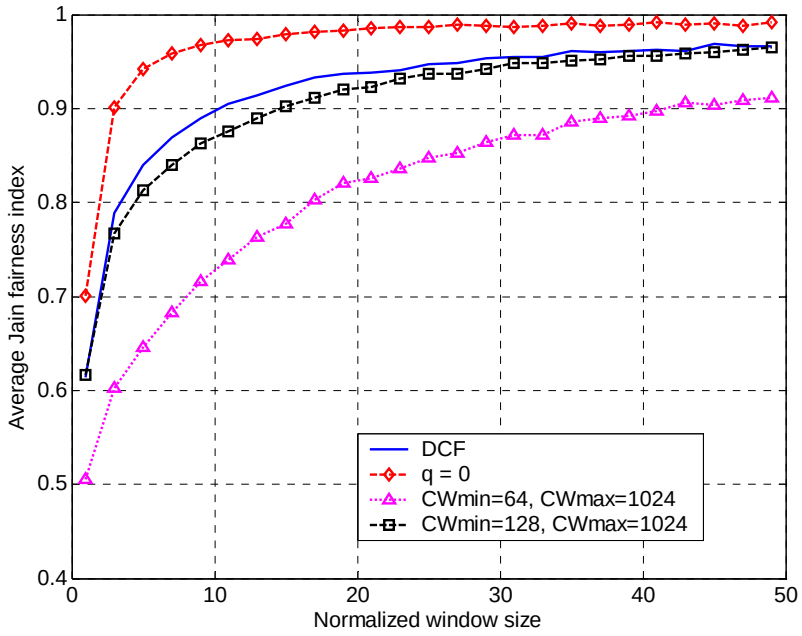

Figure 44: Fairness vs. normalized window size (n = 5)

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

In this research, we studied the problem of enhancing the backoff procedure of the DCF function in the IEEE 802.11 WLANs. The standard IEEE 802.11 WLANs are known for their flexibility, cost effectiveness and ease of use. Nevertheless, the standard's MAC layer (DCF specifically) suffers from several problems that reduce the efficiency of such networks. One of the main drawbacks in DCF is its weak support for QoS. Many applications, such as real-time applications, require QoS support and some guarantees to function properly. Another drawback is the problem of degraded throughput in high loaded situations; i.e. situations where the number of stations is high. The main factor that is responsible for this degradation is the backoff procedure employed in DCF. Our research explained these shortcomings and pointed out the main aspects of the factors causing such problems.

The IEEE 802.11 standard (the DCF function specifically) is thoroughly studied in the literature. We showed in our work the main research activities that studied the DCF function of the IEEE 802.11 standard. Some of the work in the literature studies only the standard (either theoretically or through simulations) to evaluate the main aspects of it. Other studies try to enhance the DCF as to improve functionality and solve some of the problems. Yet other studies try to introduce QoS support through differentiation or other techniques.

The DCF function of the IEEE 802.11 standard is a contention-based access mechanism. It is distributed in nature, and no single control is applied on the stations. It is the base for other services such as PCF. DCF is known for its exponential backoff procedure. This procedure is intended to help in reducing the number of collisions or their effect if they happen. However, this procedure has its own problems. In our present research, we showed these problems in detail, and we also showed some possible solutions for them.

We introduced two new algorithms that improve the functionality of the standard DCF function; namely, the *q algorithm* and the *two-stage algorithm*. These two algorithms capitalize on the need to reduce collisions and their effect if operating in congested networks. The *q* algorithm depends on counting the number of successive collisions as a metric for increasing and decreasing the CW of the backoff procedure. On the other hand, the two-stage algorithm is simpler, and its functionality does not incorporate the counting of collisions or any other statistics during run time.

We have performed extensive simulations to study and compare the performance of the protocols we have; that is, the DCF, the *q* algorithm and the two-stage algorithm. In these simulations, we used two kinds of traffic sources: saturated arrival traffic source and Poisson arrival traffic source. In the saturation state, each station has a frame ready for transmission. On the other hand, the Poisson arrival traffic source assumes an exponential distributed arrival of frames.

We defined, in our research, the main metrics and performance figures that we are measuring. These include throughput, delay and fairness. We showed the importance of each of them. Also, we defined the process of calculating each of them.

The results we obtained from the simulations are encouraging. There is an improvement in the performance in general. The throughput of the DCF is enhanced by the use of the q algorithm and the two-stage algorithm. When the q algorithm is applied with $q = 0$, we obtaine a large improvement in the throughput in high loaded situations. In low loaded situations, the throughput degrades, however. When $q > 0$, the throughput is better than DCF in general, except for $q = 3$. The delay also has improved with the q algorithm. The fairness of the q algorithm with $q = 0$ is superior. If $q > 0$, the improvement in fairness is slight. We showed a criterion and some examples for QoS support in the q algorithm where this is feasible.

For the two-stage algorithm, we obtained a good improvement in the throughput. The greatest advantage we obtained from the two-stage algorithm is the improvement of the throughput in both high loaded and low loaded situations. This is unlike DCF and the q algorithm, where each one achieves higher throughput in specific situations at the expense of degrading the throughput in other situations. The access delay is improved by using the two-stage algorithm. The fairness of the two-stage algorithm depends on the settings of the main parameters of the algorithm; namely, CW_{min} and CW_{max}. With appropriate choice, we got good fairness performance which is comparable to DCF.

6.1 Improvements and Future work

The new algorithms that we introduced in our present research showed better performance over the DCF function of the IEEE 802.11 standard. Nevertheless, in some situations, these new algorithms do not perform perfectly. This is because of many factors that we explained in the research. In this section, we are going to show some of the improvements that could be done to these algorithms. Leaving these improvements to a future study, we believe we can get greater benefit from the introduced algorithms.

Regarding the q algorithm, a very helpful enhancement that would maximize the throughput is to merge the two cases: $q = 0$ and $q = 1$. That is, if a wireless station can estimate the number of active stations in the network, a specific value of q can be chosen during run time depending on the status of the network. Namely, when $n < 10$, q is given the value zero; but when $q > 10$, q is given the value one. This would maximize the throughput of the network as shown in Figure 10. There are many techniques to estimate the number of stations in the network such as the one in [14]. Some of these techniques will give good estimation at the expense of complicating the algorithm. The complication is due to too many measurements and calculations that a station would perform during run time.

As shown in our research, the q algorithm is a good candidate for supporting QoS in wireless networks. We showed a sketch of the procedure that could be used to facilitate this, as well as some examples. A complete simulation study would show the benefits of this. The study would simulate different traffic sources and apply them to the q algorithm according to the settings shown to achieve QoS support.

We showed in our research the impact of introducing Poisson traffic arrival on the metrics of the network such as throughput, access delay and queuing delay. A tentative improvement in the research is to calculate an estimate of the number of frames in the queue at any instant of time. This is helpful for designers to estimate the maximum size of the queue of a wireless station.

As mentioned earlier, the two-stage algorithm achieves good performance in terms of throughput and fairness. As explained in our research, the two-stage algorithm depends on the choice of CW_{min} and CW_{max}. We can take the full advantage of the algorithm by introducing a mathematical technique that produces the optimal choice of these two parameters. If such a technique is found, the throughput and fairness may be controlled. The benefit of this is to maximize the throughput and fairness, or at least to maximize one of them while keeping the other at a higher value.

Like the q algorithm, a differentiation mechanism can be applied to the two-stage algorithm to facilitate QoS support. The two main parameters of the two-stage algorithm can be controlled in order to achieve this goal. Different stations can be assigned different values for CW_{min} and CW_{max}. Of course this depends on each station's need. For example, stations carrying real time traffic would be assigned values that maximize the throughput and minimize the delay. Other stations (non-real time) would be assigned other values that may result in a lower throughput or a higher delay.

When we showed the results for the fairness of the two-stage algorithm, we compared them to the DCF and the q algorithm with $q = 0$. This is because the DCF is the standard and acts as a reference. The q algorithm with $q = 0$ achieves the highest among

other values of q for the q algorithm; it is used as a typical case. Future work will compare the fairness of the two-stage algorithm to the other values of q (i.e. $q > 0$). Also, the fairness of the two-stage algorithm with greater number of stations will be considered in future work.

Fairness in general is calculated by observing and making some statistics on a sequence of frame transmissions. In our simulations, this sequence is different for each value of normalized window size. This does not harmfully affect the results. However, the ideal case is to use the same sequence of frame transmissions for all values of a normalized window size. It is expected that the results will not change dramatically. We leave this for future work.

Our simulations assumed an error-free environment for simplicity. A future study will investigate the network in noisy environments, in order to examine the functionality of the new algorithms. Another feasible and important study will consider algorithms with different transmission rates.

NOMENCLATURE

ACK	Acknowledgment
AP	Access Point
BSS	Basic Service Set
CFP	Contention Free Period
CP	Contention Period
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
CTS	Clear to Send
CW	Contention Window
DCF	Distributed Coordination Function
DFS	Distributed Fair Scheduling
DIFS	DCF Interframe Space
DSSS	Direct-Sequence Spread Spectrum
FHSS	Frequency Hopping Spread Spectrum
IBSS	Independent Basic Service Set
IEEE	Institute of Electrical and Electronics Engineers
IR	Infrared
MAC	Medium Access Control
PC	Point Coordinator
PCF	Point Coordination Function
PHY	Physical layer
PIFS	PCF Interframe Space
QoS	Quality of Service
RTS	Request to Send
SIFS	Short Interframe Space
STA	Station
TBTT	Target Beacon Transmission Time
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
WLAN	Wireless Local Area Network

REFERENCES

[1] IEEE 802.11 WG, Reference number ISO/IEC 8802-11: 1999(E) IEEE Std 802.11, 1999 edition. International Standard [for] Information Technology - Telecommunications and information exchange between systems-Local and metropolitan area networks-Specific Requirements – Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications, 1999.

[2] Q. Ni, L. Romdhani and T. Turletti, "A survey of QoS enhancements for IEEE 802.11 wireless LAN", Wireless Communications and Mobile Computing, vol. 4, pp. 547-566, 2004.

[3] H. Zhu et al., "A survey of quality of service in IEEE 802.11 networks", IEEE Wireless Communications Magazine, August 2004, pp. 6-14.

[4] S. Mangold, S. Choi, G. Hiertz, O. Klein, B. Walke, "Analysis of IEEE 802.11e for QoS support in wireless LANs", *IEEE Wireless Communications Magazine*, December 2003, pp. 40-50.

[5] S. Mangold, S. Choi, P. May, O. Klein, G. Hiertz, L. Stibor, "IEEE 802.11e wireless LAN for quality of service", *Proceedings of European Wireless (EW2002)*, Florence, Italy, February 2002.

[6] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *IEEE Journal on Selected Areas in Communications*, vol. 18, No.3, pp. 318–320, Mar. 2000.

[7] E. Ziouva and T. Antonakopoulos, "CSMA/CA performance under high traffic conditions: throughput and delay analysis," *Computer Communications,* vol. 25, pp. 313–321, 2002.

[8] Y. Xiao, "A simple and effective priority scheme for IEEE 802.11", *IEEE Communications Letters*, Vol. 7, No. 2, Feb. 2003.

[9] I. Aad and C. Castelluccia, "Differentiation mechanisms for IEEE 802.11", in *IEEE INFOCOM 2001*, Anchorage - AK, USA, April 22-26, 2001. pp. 209–218.

[10] I. Aad and C. Castelluccia, "Remarks on per-flow differentiation in IEEE 802.11", *European Wireless 2002*, Florence, Italy, February 25-28, 2002.

[11] L. Zhao and C. Fan, "Enhancement of QoS differentiation over IEEE 802.11 WLAN", *IEEE Communications Letters*, vol. 8, No. 8, August 2004, pp. 494-496.

[12] M. Barry, A.T. Campbell and A. Veres, "Distributed control algorithms for service differentiation in wireless packet networks", in: *Proceedings of IEEE INFOCOM* (2001).

[13] A. Veres, A. Campbell, M. Barry and L. Sun, "Supporting service differentiation in wireless packet networks using distributed control", *IEEE Journal on Selected Areas in Communications*, vol. 19, No. 10, October 2001.

[14] F. Calì, M. Conti, and E. Gregori, "Dynamic tuning of the IEEE 802.11 protocol to achieve a theoretical throughput limit," *IEEE/ACM Transactions on Networking*, vol. 8, No. 6, December 2000, pp. 785-799.

[15] F. Calì, M. Conti, and E. Gregori, "IEEE 802.11 protocol: design and performance evaluation of an adaptive backoff mechanism" *IEEE Journal on Selected areas in communications*, vol. 18, No. 9, September 2000, pp. 1774-1786.

[16] L. Bononi, M. Conti, and E. Gregori, "Runtime optimization of IEEE 802.11 wireless LANs performance", *IEEE Transactions on Parallel and Distributed Systems*, vol. 15, No. 1, January 2004, pp. 66-80.

[17] Q. Ni, I. Aad, C. Barakat, T. Turletti, "Modeling and analysis of slow CW decrease IEEE 802.11 WLAN", *Proceedings of the 14th IEEE 2003 International Symposium on Personal, Indoor and Mobile Radio Communication*, pp. 1717-1721.

[18] C. Wang, B. Li, L. Li, "A new collision resolution mechanism to enhance the performance of IEEE 802.11 DCF", *IEEE Transactions on Vehicular Technology*, vol. 53, No. 4, July 2004, pp. 1235-1246.

[19] J. L. Sobrinho and A. S. Krishnakumar, "Real-time traffic over the IEEE 802.11 medium access control layer" *Bell Labs Tech. Journal*, vol. 1, pp. 172–187, Autumn 1996.

[20] J. L. Sobrinho and A. S. Krishnakumar, "Quality-of-service in ad hoc carrier sense multiple access wireless networks", *IEEE Journal on Selected Areas in Communications*, vol. 17, No. 8, August 1999, pp.1353-1368.

[21] E. M. M. Winands, T. J. J. Denteneer, J. A. C. Resing, and R. Rietman, "A finite-source feedback queuing network as a model for the IEEE 802.11

distributed coordination function," *5th European Wireless Conference: Mobile and Wireless Systems beyond 3G*, Barcelona, Spain, Feb. 2004, pp. 551–557.

[22] M. Garetto and C-F. Chiasserini, "Performance analysis of the 802.11 distributed coordination function under sporadic traffic", *Networking 2005, Waterloo-Ontario, Canada.*

[23] C. Koksal, H. Kassab, and H. Balakrishnan, "An analysis of short-term fairness in wireless media access protocols," in *Proceedings of ACM SIGMETRICS, 2000*, Santa Clara, CA, June 2000, extended version available at http:// nms.lcs.mit.edu/papers.

[24] G. Berger-Sabbatel, A. Duda, O. Gaudoin, M. Heusse and F. Rousseau, "Fairness and its impact on delay in 802.11 networks", in proceedings of *IEEE Globecom 2004*, Dallas, USA, 2004, pp. 2967-2973

[25] N.H. Vaidya, P. Bahl and S. Gupta, "Distributed fair scheduling in a wireless LAN", in: *Sixth Annual International Conference on Mobile Computing and Networking*, Boston, August, 2000.

[26] S. J. Golestani, "A self-clocked fair queuing scheme for broadband applications," in *IEEE INFOCOM*, 1994.

[27] A. Banchs and X. Pérez, "Distributed weighted fair queuing in 802.11 wireless LAN," *IEEE ICC '02*, vol. 5, April 2002, pp. 3121–3127.

[28] I. Stoica, S. Shenker, and H. Zhang, "Core-stateless fair queuing: achieving approximately fair bandwidth allocations in high speed networks," in

Proceedings of ACM SIGCOMM '98, Vancouver, Canada, August 1998, pp. 118–130.

[29] R. Jain, *The Art of Computer Systems Performance Analysis*. John Wiley and Sons, 1991.

Vita

Adel A. Al-Akeel was born on September 7, 1978 in Onaizah, Saudi Arabia. In 2001, he was awarded the Bachelor of Science degree with second-class honors in Computer Engineering from King Fahd University of Petroleum and Minerals (KFUPM). In 2003, Adel began work towards a Master's degree in computer networks with the Department of Computer Eengineering at KFUPM.

Mr. Al-Akeel has worked for one year as a graduate assistant and two years as a lecturer at Qassim University. His research interests include: computer network design and simulation.